Author:
Andrew, William

Title:
Visual Biometric Processes for Collective Identification of Individual Friesian Cattle

Author:
Andrew, William

Title:
Visual Biometric Processes for Collective Identification of Individual Friesian Cattle

DEPARTMENT OF COMPUTER SCIENCE

Visual Biometric Processes for Collective Identification of Individual Friesian Cattle

William Andrew

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Doctor of Philosophy in the Faculty of Engineering.

December, 2018

To Robert,

I did publish a paper every year!

Abstract

The task of visually identifying similar objects often necessitates multiple, variant observations. As humans, we actively manipulate conditions towards differing viewpoints, object configurations and more. Within this process, an agent integrates past and present information to build understanding and inform future observations. Considering the realm of visual animal biometrics, current approaches typically operate within a paradigm of evaluating a single iteration or single image. When faced with especially fine-grained object categories however, single images may provide insufficient evidence alone. In this thesis, these intuitions are capitalised on to demonstrate that animal identification benefits from an iterative and integrative paradigm; proposing several visual animal biometric processes.

In the context of this work, the task of individual animal identification is investigated to demonstrate the advantages of enacting such a paradigm. Specifically, considering the automated and minimally intrusive identification of all individuals in a herd of Holstein Friesian or dairy cattle; a species exhibiting massively-variant coat pattern visual features, structures and alignments. The idea being to deem such features as an *individually-unique biometric entity* and perform *online herd identification via an active robot agent* in an agriculturally-relevant setting. Natural challenges arise from these intentions by virtue of intra-species visual similarities and alignments, surface deformability and occlusion, target position discovery and more.

This thesis demonstrates (in order) that the evaluation of *single* dorsal coat pattern still images for identification purposes via classical local features and representation learning provides an identification baseline. In scenarios with partial (self-)occlusion of discriminative features however, identification performance is improved upon by temporally-integrating architectures operating on image sequences of tracked individuals over time in a *passive* setting. Whilst this form of approach is sufficient across herds exhibiting little intra-population similarities, an *active identity recovery framework* is proposed next. In a realistic simulation environment it is shown that, actively navigating to viewpoint positions that reveal disambiguating features can improve upon purely passive scenarios; concluding individual cattle identification.

Next, inter-individual navigation is considered, where an agent is tasked with locating individuals in dynamically moving herds. This culminates in the finding that artificial neural networks can effectively learn herd-like spatio-temporal distributions from example. Finally, preliminary real-world experiments provide a proof-of-concept that an Unmanned Aerial Vehicle (UAV) agent can robustly discover and passively identify individual members of a small herd – combining the tasks of exploration and identification.

Altogether, this work suggests that contemporary approaches founded in deep learning in conjunction with a UAV agent utilising existing technologies can play a viable role in improving livestock welfare within the growing future of robotic and automated agriculture.

Acknowledgements

That was tough. As a result, this project would not have been possible were it not for the help and support from many people. To those people, alongside all others with involvement in this project in whatever shape or form, as I will explicitly mention here, I say *thank you*.

First and foremost, I express utmost gratitude to my supervisors: Dr Tilo Burghardt and Dr Colin Greatwood. Tilo, your persistent enthusiasm, passion and guidance throughout my Masters dissertation and now this PhD thesis has been remarkable and I look forward to working closely with you again in the near future. Colin, this project would have been impossible were it not for your extensive knowledge and pragmatism, particularly for the final set of experiments.

Alongside my supervisory team, there is an expansive network of people responsible for making this project possible to begin with, as well as providing support along the way. Principally, I think Professor Arthur Richards for realising and directing the vision of spearheading robotic PhD research in the UK in founding the Future Autonomous and Robotics Systems Centre for PhD Education (FARSCOPE) CDT – a project made possible by EPSRC funding. As part of the centre, I thank all past and present administrative staff. In particular, Dr Sally Birse, Michael Secker and Olga Nossenko. Special thanks goes to Gareth Griffiths for technical support throughout the project. Thanks also to Professor Becky Whay and Kate Robinson for making experiments with live cattle possible, as well as Dr Sion Hannuna and Dr Neill Campbell for capturing and processing some of the data used in the early stages of this work.

Next, I thank the unfaltering support from my family throughout my undergraduate and now postgraduate studies here in Bristol. In particular, my father for saving the day at a critical point during the conduction of final flight experiments, my mother for the gladly received cocoa and hops-based support, and my brother for providing welcome (and often musical) distractions from work. Special mentions of gratitude go to my two grandmothers in Paris and Braintree alongside all other members of our family. I especially thank my late Grandfather, Professor Robert Morfin – to whom I dedicate this thesis – for inspiring me to follow in your footsteps as a person and academically.

I finish by expressing thanks to close friends, with special admiration and gratitude to my best friend, Kate. Thanks for putting up with me these past couple of years.

Declaration

This dissertation is submitted to the University of Bristol in accordance with the requirements of the degree of PhD in the Faculty of Engineering. It has not been submitted for any other degree or diploma of any examining body. Except where specifically acknowledged, it is all the work of the Author.

William Andrew, Tuesday 26th February, 2019

Contents

List of Publications

In order of publication date:

1. W. Andrew, S. Hannuna, N. Campbell, T. Burghardt. Automatic Individual Holstein Friesian Cattle Identification via Selective Local Coat Pattern Matching in RGB-D Imagery. In *IEEE International Conference on Image Processing (ICIP)*, pages 484-488, 2016. https://doi.org/10.1109/ICIP.2016.7532404.
2. W. Andrew, C. Greatwood, T. Burghardt. Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning. In *IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 2850-2859, 2017. https://doi.org/10.1109/ICCVW.2017.336.
3. W. Andrew, C. Greatwood and T. Burghardt. Deep Learning for Exploration and Recovery of Uncharted and Dynamic Targets from UAV-like Vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1124-1131, 2018. https://doi.org/10.1109/IROS.2018.8593751.

List of Abbreviations

ANN	Artificial Neural Network
API	Application Programming Interface
ASIFT	Affine Scale-Invariant Feature Transform
AuC	Area under Curve
BID	Bovine Identification Document
BGR	Blue Green Red
BPTT	Back-Propagation Through Time
CAA	Civil Aviation Authority
CAD	Computer-Aided Design
Caffe	Convolutional Architecture for Fast Feature Embedding
COCO	Common Objects in Context
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DoF	Degrees of Freedom
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DRQN	Deep Recurrent Q-Network
DQN	Deep Q-Network
ECEF	Earth-Centered Earth-Fixed
EER	Equal Error Rate
ENU	East North Up
FCN	Fully Convolutional Network
FLU	Front Left Up
FRD	Front Right Down
FSM	Finite State Machine
GCS	Geographical Coordinate System
GIS	Geographical Information System
GPS	Global Positioning System

GPU Graphical Processing Unit

GPGPU General Purpose Graphical Processing Unit

GRU Gated Recurrent Unit

GUI Graphical User Interface

ILSVRC ImageNet Large Scale Visual Recognition Challenge

IoU Intersection over Union

KCF Kernelised Correlation Filter

LAIDN Local Active IDentification Network

LAN Local Area Network

LBP Local Binary Pattern

LIFO Last In First Out

LiPo Lithium Polymer

LRCN Long-term Recrrent Convolutional Network

LSTM Long-Short Term Memory

mAP mean Average Precision

MDP Markov Decision Process

MLP Multi Layer Perceptron

MOT Multi-Object Tracking

MSE Mean Square Error

NBV Next Best View

NED North East Down

NN Nearest Neighbour

OES On-board Embedded System

ORSA Optimized Random Sampling Algorithm

PCA Principal Component Analysis

PID Proportional Integral Derivative

POMDP Partially-Observable Markov Decision Process

P-RNG Pseudo-Random Number Generator

RAM Random Access Memory

RANSAC Random Sample Consensus

RBF Radial Basis Function

R-CNN Regional Convolutional Neural Network

R-FCN Region-based Fully Convolutional Network

RGB Red Green Blue

RGB-D Red Green Blue Depth

RNG Random Number Generator

RNN Recurrent Neural Network

ROC Receiver Operating Characteristic

RoI Region of Interest

ROS Robot Operating System

RPN Region Proposal Network

RPY Roll Pitch Yaw

RTK Real-Time Kinetic

RTRL Real-Time Recurrent Learning

SAR Search and Rescue

SIFT Scale-Invariant Feature Transform

SDK Software Development Kit

SGD Stochastic Gradient Descent

SoC System on Chip

SLAM Simultaneous Localisation and Mapping

SSH Secure Socket Shell

SURF Speeded Up Robust Features

SVM Support Vector Machine

TSP Travelling Salesman Problem

UAS Unmanned Aircraft System

UAV Unmanned Aerial Vehicle

UIN University Investigation Number

VOC Visual Object Classes

VTOL Vertical TakeOff and Landing

WGS World Geodetic System

W-LAN Wireless Local Area Network

XML Extensible Markup Language

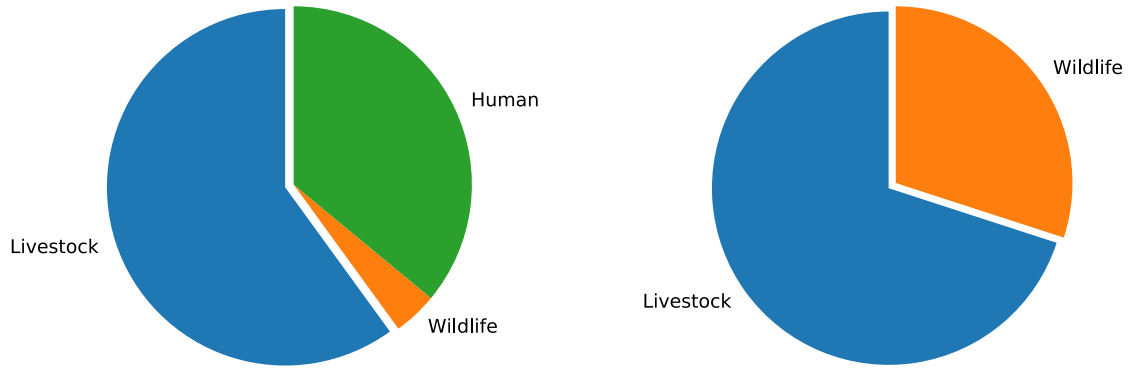
YOLO You Only Look Once

Chapter 1

Introduction

1.1 Motivation

Motivation of the research presented within this thesis is justified by staggering statistics; of *all* mammal biomass[1] on earth today, just 4% is wildlife [22]. Of the overwhelming majority remainder, 36% is human and, therefore, *60% of all mammal biomass is livestock*. This breadth of human-driven consumption also extends to birds, where 70% of their biomass is chicken and other poultry, and just 30% is wild (refer to Figure 1.1 for visual context). The effect of which is environmentally devastating; being one of the largest contributors to greenhouse gases, loss in biodiversity and consumption of arable land [296]. Looking towards mammal husbandry specifically – being largely comprised of bovine and swine [22] – it forms the vast majority of living mammal biomass on the planet. Considering this fact, dedicating research efforts towards improving the welfare of livestock is certainly effort well spent. This thesis consequently directs work towards improving the welfare of cattle mammals, specifically the Holstein Friesian species.



(a) Mammals *(b) Birds*

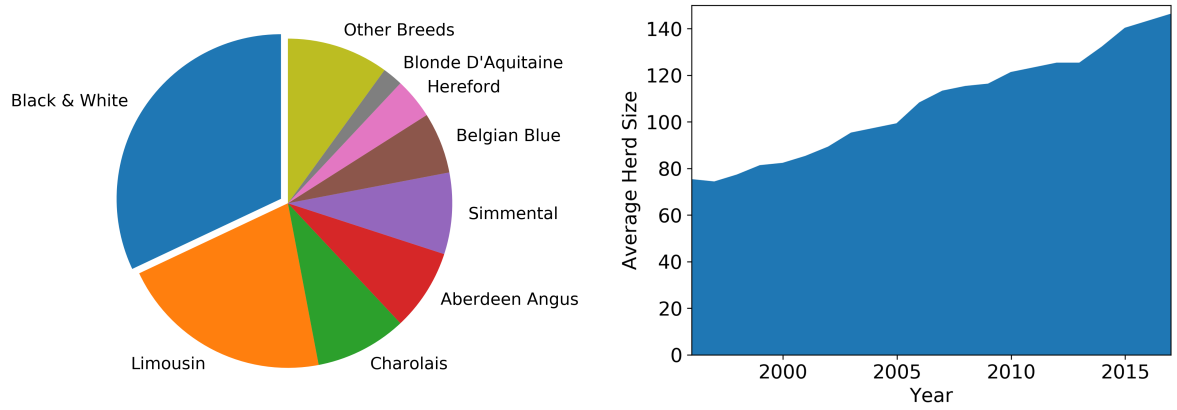
Figure 1.1: ***Animal Biomass.*** *The estimated proportion of contemporary animal biomass on earth for (a): mammals and (b): birds [22]. In both classes of animals, livestock constitutes the vast majority; cattle and pigs for mammals, chicken and others for birds [22].*

1.1.1 Holstein Friesian Cattle

Holstein, Friesian and Holstein Friesian cattle – more commonly referred to as 'dairy cows' – are the highest milk yielding breeds in existence [304]. Consequently, they represented the majority of cattle

[1] 'Biomass' Oxford dictionary definition: "the total quantity or weight of organisms in a given area or volume".

breeds in the UK across multiple years [227, 67] and their numbers per-farm have gradually risen year by year, as visualised in Figure 1.2. On an international scale, the bovine industry is economically significant globally, producing over $3.5 billion in beef and veal exports in the US in 2010 alone [277]. Current export requirements and consumer demand require visual identification of livestock [39], which is legally mandated in some countries/unions [237, 231]. Frameworks developed towards this requirement largely couple national databases and ear-tagging [140, 43, 283].



(a) Cattle breed distribution *(b) Average UK dairy herd size*

Figure 1.2: ***Cattle Distributions.*** *Distributions of cattle breeds and herd sizes within Great Britain and the United Kingdom. (a): illustrates the 8 most common breeds of cattle in Great Britain in 2008 [67]. The 'Black & White' category collectively contains the Friesian, Holstein Friesian, British Friesian and Holstein breeds as they are difficult to distinguish during census. In total there were nearly 9 million registered cattle in Great Britain in 2008, approximately 3 million of which were 'Black & White' [67]. (b): shows the average size of dairy herds per farm within the UK doubling over the past two decades [6, 5].*

The majority of individual bovine identification revolves around manual (e.g. ear tags, branding [183], tattooing [39]) or electronic labelling [264]. Meanwhile, the benefits of identification ability contribute towards improved cattle welfare through health and social monitoring, traceability, control of disease outbreak and more [124, 292, 39, 46]. Commonly throughout agriculture, cattle, sheep and others are manually identified via the use of ear tags containing a unique identification number alongside additional descriptive components [317] (see Figure 1.3). For European cattle, two ear tags (for redundancy) and a Bovine Identification Document (BID) are mandated by European Parliament regulation 820/97 [237]; which was later repealed and updated by regulation 1760/2000 [238]. Concern however, has been voiced about the success of manual tagging identification methods [80, 323]. Primarily, the tag is subject to being lost or damaged beyond recognition [96] and thus alone, may be insufficient in providing long-term identification. From a welfare standpoint, animal well-being is called into question, as tags may permanently damage or alter an animal's ear [151, 79].

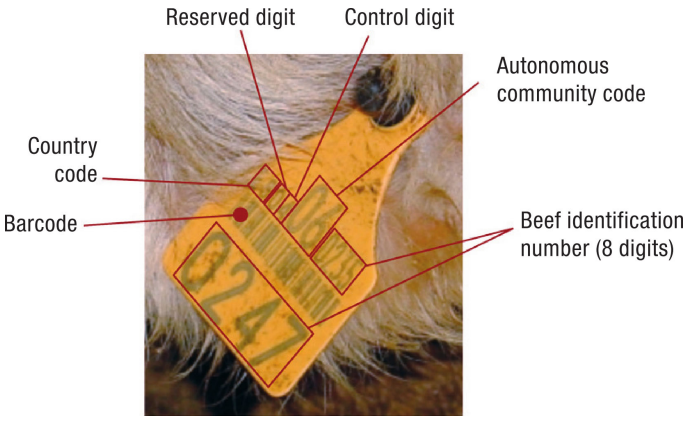


Figure 1.3: ***Bovine Ear Tag.*** *Example of an EU-mandated bovine ear tag [237] with information annotations. Image credit: Velez, J. F. et al. [317].*

Figure 1.4: ***Freely Grazing Holstein Friesian Cattle.*** *Visible in the image are the individually-unique dorsal coat patterns exhibited by the species that are considered in this thesis to be a biometric entity that can be exploited for individual animal identification.*

Accordingly, this thesis proposes to exploit naturally occurring features in Holstein Friesians for identification. As is exemplified in Figure 1.4, members of the species exhibit individually-unique dorsal[2] coat patterns. Coats are comprised of black and white structures and distributions that are proposed here to be a biometric entity. Under this assumption, animal identity can be inferred from appropriate two-dimensional imagery. As a consequence, *automated* and *minimally intrusive* identification of Holstein Friesians is made possible. In doing so, promising to contribute beneficially towards improving animal welfare, in rendering physical tagging methods to be unnecessary. A factor which, given the aforementioned significance of bovine mammals and biomass in the world, we as humans have a responsibility towards addressing. This thesis, therefore, seeks to investigate *visual biometric processes for autonomously identifying individual Holstein Friesian cattle based on imagery of dorsal coat patterns alone*.

Aside from individual identification, lameness detection and gait analysis form a large area of modern dairy husbandry research – an area which could be assisted by automated visual identification. This volume of work is motivated by the fact that mastitis and lameness are the most frequently occurring diseases within dairy cattle [169]. As a result, methods and algorithms applied in reality can ultimately yield preventative measures in signalling early signs of lameness [214, 149], a problem which causes milk production loss [324, 114] and potential early culling [87]. Automations in this area of research reside in vision-based [293, 247, 318, 148] and sensor-based [253, 316, 240] methods to assess animal gait using various scoring metrics [295, 252].

1.1.2 Collectivity, Acquisition and Agency

Building upon the automatic and minimally-intrusive identification of Holstein Friesians, improvements to their welfare can go further. In alignment with biological, veterinarian, behavioural and welfare research interests is the goal of studying cattle herds in real agricultural environments. Studies analysing

[2] 'Dorsal' Oxford dictionary definition: "on or relating to the upper side or back of an animal, plant, or organ"

appropriate data could inform on emergent intra-population social hierarchies and constructs [313, 166, 243] as well as overall welfare (e.g. physical, societal) of the herd and its constituent individuals [294], grazing patterns [116, 117] and more. An example of data permitting such analysis could consist of recorded individual identities and their positions over the course of a social monitoring experiment. In this thesis, acquisition of such data is demonstrated to be autonomously attainable with combination of the proposed techniques. This involves the use of a robot agent, specifically an Unmanned Aerial Vehicle (UAV) quadrotor, able to fly freely in six Degrees of Freedom (DoF) and importantly, stably hover at a defined position.

Usage of an aerial platform is reinforced by the identification problem itself; in exploitation of exhibited dorsal features, appropriate imagery containing such a viewpoint must be obtained. Utilisation of a flying robot, therefore, permits such aerial viewpoints to be readily captured by an on-board camera. In addition, a UAV agent allows aforementioned herd monitoring goals to be realised in an autonomous setting. The robot can move freely about the environment, focusing attention on identifying a particular individual and subsequently, move on to find new individuals to be identified. Achieving this form of target discovery by environment exploration is costly when UAV flight times are limited by battery life. Attempting to minimise this cost implies discovering new individuals to be identified more efficiently. Consequently, this thesis will *investigate efficient environment exploration strategies towards discovering unknown targets.*

1.1.3 Fine-Grained Recognition in Animal Biometrics

Fine-grained visual recognition tasks deal with categorising objects with higher levels of specificity than their basic class. As a simple example, the base class of 'bird' for an object 'eagle' is easily visually determinable through modern object detectors [171]. However, differentiating 'eagle' from 'hawk' poses difficulty and requires more attention to fine-grained details. More formally;

> *"Whereas visual recognition research is mainly focused on two very different situations; distinguishing between basic-level categories (category recognition) or recognising specific instances (instance recognition), developing algorithms for automatically discriminating categories with only small subtle visual differences (fine-grained recognition) is a new challenge."*[3]

The difficulty is that these discriminative details are often overwhelmed by small changes to viewpoint, object pose and location in imagery. Recent advances in deep learning has spearheaded state-of-the-art results on benchmark datasets involving fine-grained object categories with base classes including flowers [284], birds [348, 40, 351, 98], dogs [161, 101, 281], cars [170, 197], and more. These results indicate significant improvement on hand-crafted feature representations in the ability to learn good discriminative deep convolutional features [284] – a crucial property of fine-grained recognition algorithms.

This work is no different; (a): cattle generally need to be detected and located in relevant imagery (e.g. 'Cow': base class) and (b): individual cattle identities need to be determined (e.g. 'Amanda': specific identity class). The difficulty lies in the fact that, (1): coat pattern features exhibited by individuals can be incredibly similar in structure, shape and positioning on the body, whilst simultaneously (2): there can be massively-variant patterns amongst a population. Additionally, Holstein Friesian coat patterns provide disruptive camouflage [297] that relies on high contrast binary colouration that can pose a challenge in object localisation [326], relative viewpoint determination and of course, individuals with similar markings. Examples of visual challenges are given in Figure 1.5, where row ***(a)***: depicts pairs of differing individuals with similar markings and similar spatial distributions of markings over their bodies.

[3] Quote from Dr Erik Rodner – http://erodner.github.io/researchpage/finegrained.html

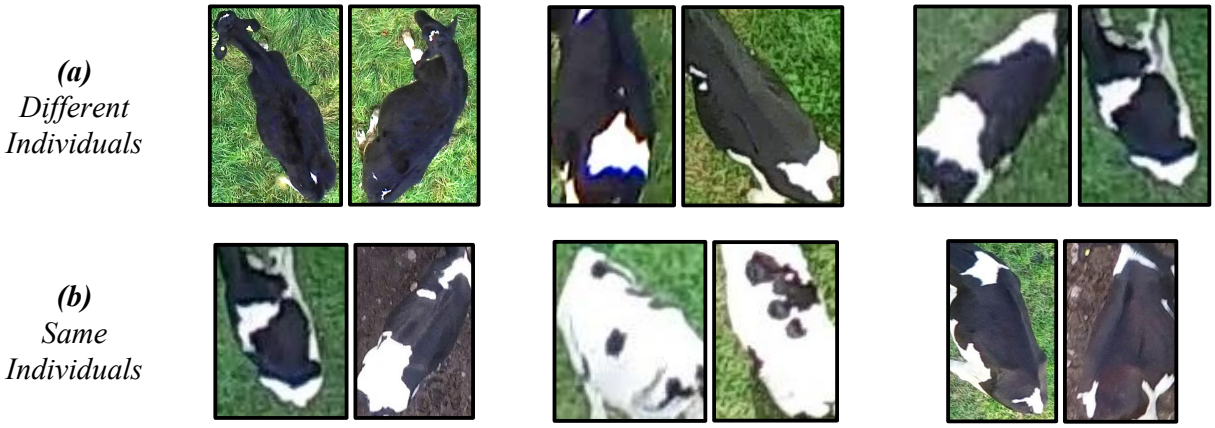


Figure 1.5: ***Fine-Grained Recognition Task.*** *Examples of identification difficulties present in this work on fine-grained recognition of individual cow identity classes. Pairwise examples are given where image pairs contain* ***(a)****: different individuals and* ***(b)****: the same individuals.*

As a result of particular individuals exhibiting many visual similarities, *a single query image may provide insufficient evidence to determine the identity of the subject cow altogether*. Equally, a model evaluating a single image may be confident in an identity to some unsatisfactory extent. Methods operating on single images, as is common for fine-grained recognition methods [98, 197, 351], are therefore, fundamentally flawed in this specific domain. The extent to which this paradigm of single-image evaluation is successful on the task of cattle identity estimation is investigated fully in Chapter 3, using hand-crafted and learnt features. There is consequently, a justified need for an approach that integrates information from multiple variant observations to disambiguate these cases of identification problem; an *identification process*. This fact has been used for advantage in the area of face recognition [359, 4], where approaches consider observations separately and form a voting or scoring mechanism, fuse observations into a single array/matrix [333], observations form the basis of a probability density function [282] and more [188].

Further to the challenges on the evaluation of single images on fine-grained categories; salient discriminative features are often subtle, and thus, models are sensitive to small perturbations in viewpoint, object pose, etc. These are exemplified well in Figure 1.5, where row ***(b)***: highlights pairs of the same individuals that are challenging to associate from viewpoint and pose changes. Such perturbations are caused and exacerbated by variational error from predecessor models (e.g. object/region detector), agent viewpoint control and object pose autonomy (cattle are live, moving animals). The advantage then, in evaluating multiple images to build a representation of identity, is that the likelihood of observing discriminative features increases. Put differently, it becomes increasingly likely that some observation will contain none of these problems caused by perturbations. This is true even when an agent passively observes live objects like cattle; it takes no actions, but object pose change (e.g. walking, looking) reveals salient features. This intuition is capitalised on in this thesis, with Chapter 4 exploring *multi-observation fine-grained identification in a passive setting*.

People Re-Identification

The very same challenges that are exemplified in Figure 1.5 form the basis of the difficulties surrounding the well studied area of people re-identification; the task of associating persons of interest across different camera locations. The core challenges there being: there may be little template knowledge of individuals (e.g. single viewpoints) and other cameras may obtain new object viewpoints altogether, the problem can be incredibly open set (there may be no template knowledge of all individuals), target persons may be heavily occluded (e.g. crowded environments), individual articulation or pose is highly variable and much more [113].

Research efforts in this area have traditionally involved hand crafting low level viewpoint and lighting-invariant features [94, 354, 191, 335] or employing metric learning [48, 352, 191, 347]. Recently however, Convolutional Neural Network (CNN)-based solutions have shown success in the area by jointly learning feature representations and comparison metrics in an end to end fashion [72, 49, 7, 320] that has seen human-level performance surpassed [350]. One such example jointly learns feature embeddings for image pair classification as well as verification via a Siamese network [358].

In relation to this thesis, proposing a collection of identification processes for individual cattle, there is rarely a need for verification (i.e. suggesting whether two images contain the same individual or not). In fact, in time-sensitive scenarios (particularly when operating identity inference onboard a UAV), verifying many image pairs is perhaps too costly, as Section 3.2 will establish. Instead, single images are frequently acquired to be classified, meaning that a single feature embedding can be learnt over the set of categories, much like some approaches to people re-identification [355, 356, 357]. This does however, carry the assumption that the set of classes is closed, whereas verification models can suggest newly-encountered categories.

1.1.4 Active Vision

When a sequence of passively obtained images still contains insufficient evidence for an individual's identity, despite small variations amongst constituent frames, an alternate approach is justified. An example of such an approach falls into the category of active vision, whereby an agent is actively involved in manipulating intrinsic and extrinsic sensor parameters for some vision task or *process*. The need for this case is exemplified in Figure 1.6. In the left-hand image, the metal cup and mug are challenging to differentiate. It's only when the mug's handle becomes visible from a significant change in object pose that one can confidently state the right object in both images is a mug. This significant change is unlikely to occur in passively obtained sequences of images. As is the case here, active manipulation of observation parameters is necessitated.

In relation to the task of individual cattle identification, there may be circumstances where observed dorsal features alone provide insufficient information gain. Only by observing a very particular viewpoint or particular set of viewpoints can identity categories be disambiguated, thus necessitating an active approach. In this work, the agent (a quadrotor UAV) is able to move freely in three dimensions to realise new object viewpoints in order to observe discriminative features; performing an *iterative and active identification process*.

Figure 1.6: ***Disambiguating Viewpoints.*** *(Left): the metal cup and mug are challenging to differentiate in this configuration. (Right): the presence of the mug's handle – a visual feature crucial for visual identification purposes – as a result of object manipulation now permits trivial object differentiation. In the case that this single viewpoint provides maximal information gain on possible object categories, it is referred to as the "canonical viewpoint" [78].*

1.2 Research Objectives

In this section, principal objectives of the entire thesis are collected and summarised from the aforementioned motivation of research:

- Characterise the extent to which dorsal coat patterns exhibited by Holstein Friesian cattle are individual or unique across small population sizes as an implicit part of the following tasks.
- By assuming coat patterns to be a biometric entity, investigate in how far the paradigm of *evaluating a single image* is effective on fine-grained identity categories (individual Holstein Friesian cattle) from a standard aerial viewpoint of dorsal features.
- Demonstrate that supposed model belief in a class and resulting accuracy can be reinforced and improved by multiple observations over time, whereby inter-sample variations occur naturally without agent influence; a *passive identification process*.
- Involve an agent actively within the identification process, whereby particular viewpoints are sought on a per-individual and per-discovery basis to disambiguate object observations as efficiently as possible; an *active identification process*.
- Research robust environment exploration strategies via simulations for an agent with local sensing capabilities towards *efficiently discovering the positions of targets in a closed domain*.
- Fuse developed components under a common framework executing on-board a real UAV agent with respect to the computational and physical constraints imposed by use of such a mobile robot platform.
- Deploy the flying robot to critically assess the performance of the proposed methodologies in reality by conducting *online census of a live cattle herd*.

1.3 Thesis Outline and Contributions

In this section, an outline is given for this thesis, alongside the core contributions of each chapter. The remainder of this thesis consists, firstly, of concluding the opening chapters with **Chapter 2 – Background**. This chapter provides appropriate knowledge designed to give the reader an understanding of the core concepts employed in this work, all whilst aligning relevant seminal and contemporary literature with the intentions and objectives of this thesis. Second, the five core work chapters are outlined as follows:

- The first three work chapters outline varying visual biometric processes for the task of an agent visually inferring the identity of a *single* individual cow it is positionally local to. The described methodologies, with increasing levels of agency and complexity, exploit the uniqueness of individually-exhibited coat patterns for Holstein Friesian cattle.
  - **Chapter 3 – Single Frame Identification**: details the extent to which a traditional, single-frame evaluation paradigm operates well on the task of fine-grained individual cattle identification by visually exploiting coat patterns. This demonstration is split into two approaches; (a): extracting and matching local hand-crafted features and, (b): learning representative features per-individual with an Artificial Neural Network (ANN). The employed approaches suggest that the proposed methodologies are well-suited towards extracting or learning discriminative features from fine-grained object categories.
  - **Chapter 4 – Passive Multi-Frame Identification**: investigates the effect of exposing a visual identification model to multiple, varying iterations comprised of viewpoint change, illumination variation, object pose variability and more. Image sequences are acquired *passively* (with no agency) and presented as input, such that salient spatio-temporal identification features are integrated through time. This form of approach is demonstrated to be beneficial to identification accuracy and confidence, validating the proposed iterative identification pipeline. Within this process, a species-wide detector demonstrates robust performance

across relevant imagery through off-the-shelf components.

  - **Chapter 5 – Simulated Active Multi-Frame Identification**: proposes a unified framework for the performance of *active* identification with a UAV-based agent. Within this, the employed architecture automatically extracts a sequence of observations or viewpoints that satisfactorily identify the target as quickly as possible. This per-individual and circumstantial behaviour is then replicated online by the agent upon discovery of a new cow to be identified in a realistic three-dimensional simulation environment with difficult synthetic identification scenarios.

- This next chapter deals with an agent autonomously navigating an uncharted environment with a target recovery objective. Specific to the objectives of this thesis, efficient exploration strategies are needed for finding (new) cattle to be identified using the UAV, sandwiching local individual identifications.
  - **Chapter 6 – Simulated Inter-Individual Navigation**: looks towards autonomously yielding effective exploratory strategies for a flying robot moving in a horizontal two-dimensional plane. The goal being to discover the positions of targets distributed in an unknown environment as quickly as possible – a class of the robotic Search and Rescue (SAR) problem. Learning by example in a supervised fashion, the proposed model is able to reliably discover a wide variety of target distribution classes, including dynamic moving targets. This is achieved by a dual-stream architecture learning features from (a): local visual sensing combined with (b): a global representation of positional history when trained against optimal navigation decisions from travelling salesman solutions. This application of a supervised learning-based approach is demonstrated to reproduce exploratory efficiency well in an online setting, outperforming baselines including a popular unsupervised reinforcement learning algorithm.

- The final work chapter of this thesis deals with providing a proof of concept that the algorithms and methodologies proposed earlier operate successfully in a real and live setting using a mobile flying robot agent.
  - **Chapter 7 – Proof-of-Concept: Real-World Herd Individual Identification**: gives findings for experiments operating with a live cattle herd and real-time processing on-board a UAV agent. Within this task, and for the first time, environment exploration and passive iterative individual identification are combined under a common framework. Details are given on the employed hardware setup as well as the extensive work and considerations surrounding the experimental setup for a flying robot. The core contribution lies in validating earlier-proposed algorithmic choices in a live setting.

To finish, **Chapter 8 – Conclusion** provides concluding remarks concerning the entire thesis alongside a summary, noteworthy points of discussion and comments about possible future avenues for work continuation.

Chapter 2

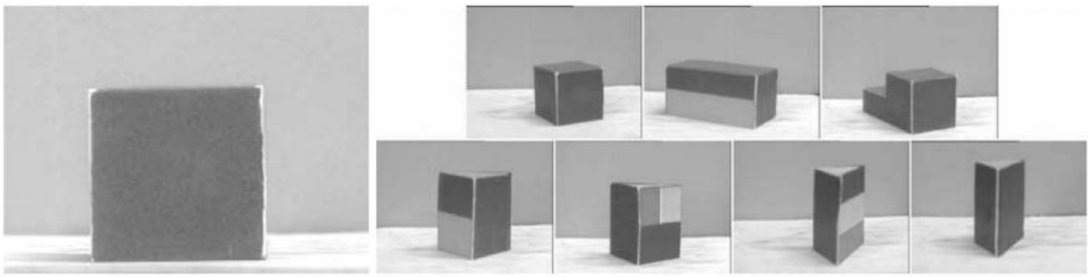
Background

2.1 Chapter Overview

This chapter introduces the concepts, ideas and related work aligned with the goals of this thesis. This chapter is therefore designed to give the reader a fundamental understanding of the underlying conceptual ideas, seminal related work and otherwise necessary to follow the remainder of this thesis, concluding the opening chapters. In addition, this chapter also discusses appropriate literature, as well as the extent of original contribution in that area.

2.2 Active Object Recognition

Active object recognition refers to the problem of visually inferring object classes from a series of observations that are actively chosen or parametrised by an agent [328]. This form of approach is justified in application-specific circumstances whereby a single image from some viewpoint does not contain sufficient features to discriminate an object from all others unambiguously (Figure 2.1 illustrates an example of this case). The realisation of new, suggested observations is completed either by manipulation of the object itself (e.g. a robot gripping the object rotates its wrist [42]) or, changing the camera viewpoint (an active sensor). In the case of this thesis, the latter is implemented – the UAV agent moves freely about the cow being identified to obtain new object viewpoints.



(a) ***(b)***

Figure 2.1: ***Active Object Recognition.*** *Justification of the need for active object recognition in relevant applications. For the viewpoint shown in* ***(a)****, this could correspond to any one of the possible object classes shown in* ***(b)****, only by actively changing viewpoint or manipulating the object can its actual class be inferred. Image credit : Roy, D. S. et al. [267].*

Seminal work in this well-studied area originates from Wilkes, D. & Tsotsos, J. K. employing a robot arm-mounted camera to differentiate origami models by realising an object-wise standard viewpoint that allows the problem to be cast as a feature matching problem [328]. Whilst this form of approach is robust

to initial viewpoint variation, it does not integrate information from multiple observations, as opposed to the proposal in this thesis. Since then, relevant works have used model-based approaches with varying view-based object representations using parametric eigenspaces [38, 37], aspect graphs [118, 265] and others [276], as well as part-based object representations [71, 266]. In addition, reinforcement learning has been used to learn disambiguative viewpoints via sensorimotor control [235], this is explored further in Section 2.3.4.

2.2.1 Next Best View

Within the realm of active vision lies the notion of Next Best View (NBV), in which the goal is to determine a good, ordered sequence of image viewpoints towards some objective, typically to iteratively build a complete representation of a scene or object. The application of such algorithms reside in automatic 3D surface/environment reconstruction [21, 246, 145, 329], environment mapping [2, 32] and exploration [31] and even 6D object pose estimation [76]. The core intentions being to (a): generate more efficient trajectories than simply exhaustively combining all viewpoints, especially since single iterations (evaluation of a particular viewpoint) may be costly, and (b): focus viewpoint attention on particularly complex regions (as can be seen in Figure 2.2). Broadly speaking, approaches may be categorised into surface-based, volume-based or global methods [279].

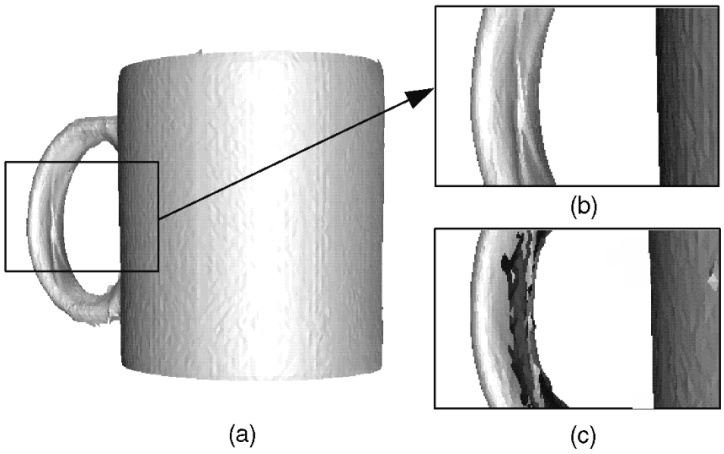


Figure 2.2: ***NBV vs. Regular Interval Scanning.*** *The surface acquisition of a mug $(a), (b)$: using a NBV algorithm and (c): without using one. For (c); the model was reconstructed from eight range images at regular intervals, where the inside of the handle was not fully reconstructed. In contrast, use of a NBV planner is qualitatively complete in this area, again across eight range images. Image credit: Pito, R. et al. [244].*

Seminal work in the area was proposed by Connolly, C. I. in 1985 [59] in formulating a volumetric algorithm. The method fits a sphere to marginally encompass and to be centred about the target object, where possible camera views are evenly distributed longitudinally and latitudinally on the surface of the sphere. A recursively subdivided octree [210] is superimposed on the object in three dimensions with three possible vertex states: empty, unseen and occupied. Next best views are progressively suggested based on viewpoints that intersect the largest unseen area.

Classical NBV problems are well aligned with the active individual identification intentions of this work; a freely moving camera (the UAV-based agent with a gimbal-mounted camera) iteratively builds understanding of an object (its identity) by progressively selecting viewpoints with respect to some objective function (satisfactorily identifying the individual as quickly as possible). The principal separation is that NBV algorithms generally act upon little or no a priori knowledge; the algorithm occurs entirely online. This is possible in applications where satisfying the objective function isn't object class-specific, however in this work, this isn't the case. Put differently, next best views identifying individual x are likely not to apply to individual y as a result of differing coat pattern markings.

In this work, a model-based graph theory approach is proposed with a distinct training phase to build a set of object class-specific behaviours. To begin, a domain-specific abstraction is formed in modelling objects (individual cattle) as a cuboid with five visible sides (e.g. top, front, back, left, right, see Figure 2.3), each with an associated optimal viewpoint. This operates upon the fair assumption that intra-population

variation occurs only amongst individual coat patterns or put differently, body sizes and proportions are consistent across a population. In an offline pass, $5! = 120$ viewpoint orderings are exhaustively searched for each individual in a population towards extracting an agent trajectory – a set of next best views – that identifies the individual whilst minimising the combination of (a): distance travelled and (b): number of viewpoints (iterations). This individual-wise behaviour is then recreated online by the agent with respect to the viewpoint from which a target (to be identified) was discovered – for full implementation details, refer to Chapter 5.

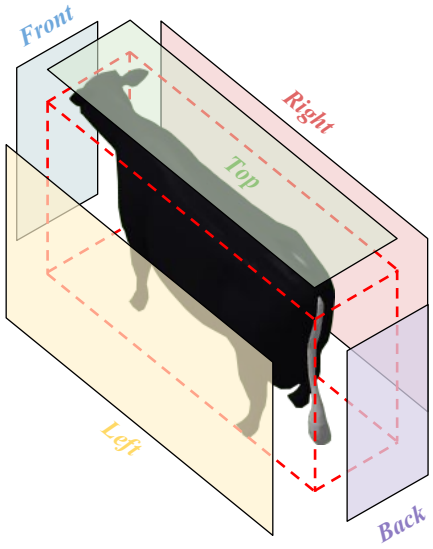


Figure 2.3: ***Cow Model Abstraction.*** *The employed parts-based representation/abstraction of cattle used in this work for active individual identification. A three dimensional cuboid is fitted to spatially enclose the object. Each of five visible faces (e.g. 'top', 'front', 'back', 'left', 'right') are attributed a static viewpoint which optimally views that part with respect to UAV flight constraints (e.g. minimum height above ground/cattle).*

2.2.2 Viewpoint Fulfilment via Object Pose Estimation

To realise a new viewpoint suggested by a NBV planner, the agent has to be aware of its spatial relationship with the subject object. As such, an estimate of the object's pose needs to be determined. Estimating three-dimensional object pose from two-dimensional images is intrinsically difficult. Established methods lie in point cloud-based approaches [189, 9] requiring additional Red Green Blue Depth (RGB-D) sensing or via approximative depth images [337, 332, 286]. However, here, housing additional sensing components on-board the UAV is costly in power consumption, weight, and space. As a result, monocular Red Green Blue (RGB)-based algorithms [203, 13] are more appropriate in this context.

This work assumes object geometry to be similar across all classes, an assumption that was applied in the model formulation for active cow identification (refer back to Section 2.2.1 and Figure 2.3). This is motivated by the domain; all cows are generally of the same shape/scale, given similar ages across a population. The result is that object scale in an image directly indicates agent-target distance. When combined with knowledge of camera pitch and yaw, information that is readily available from the UAV's sensors, this work proposes this basis to be sufficient to accurately learn and infer an agent-relative vector between the two. Fusion of these two inputs is performed implicitly in a single deep network that is trained end-to-end. This area of work is specific to Chapter 5 on active iterative identification, where it is described fully in Section 5.4.2.

2.3 Artificial Neural Networks

Artificial Neural Networks (ANN) have played a vital role in recent successes in natural language processing [112], computational chemistry [100, 111], robotics [139, 213], and more relevantly, computer vision [11]. This surge in popularity has been fuelled by deep, many-layered network architectures facilitated by recent improvements in General Purpose Graphical Processing Unit (GPGPU) technologies [171]. The result of which has yielded incredible successes at a vast array of tasks, even famously involving concretely beating the best human player at Chinese board game GO [287, 288], long thought to be computationally intractable due to game complexity [225].

Drawing biological inspiration [338, 3] and, as the name would suggest; ANNs artificially model arrays or tensors of individual neurons in some meaningful hierarchical arrangement or network. Figure 2.4 illustrates a simple example of such a network, whereby, a single neuron y_j (a vertex) is activated if the weighted sum of all predecessor neurons x_i (with edge weight w_{ij}) satisfies some activation function:

$$y_j = \sigma\left(w_o + \sum_{i=1}^{n} w_{ij} x_i\right), \tag{2.1}$$

where σ, w_0 denote the activation function (e.g. linear, hyperbolic tangent, sigmoid) and the bias to its input, respectively. In this form of forward evaluation (forward-propagation, left to right), the network *infers* knowledge from a given input. However, the benefit in such a design lies in the model's ability to learn neuron parameters (weights and biases) from example. Training the network in this supervised form is achieved via the back-propagation algorithm (right to left)); where the gradient of an error function is computed to update (typically randomly) initialised weights according to some learning rate [263].

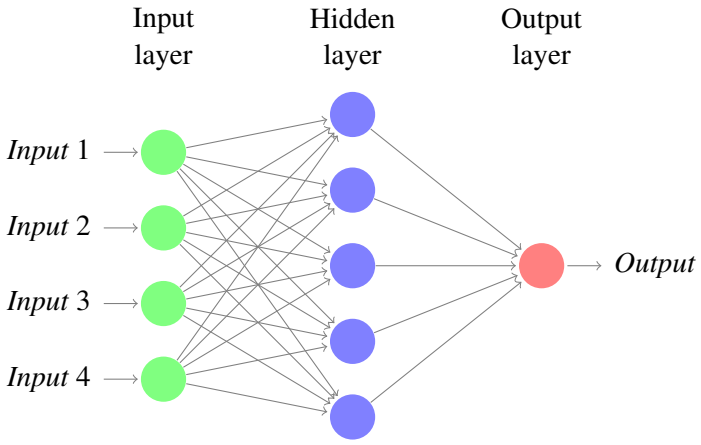


Figure 2.4: ***Fully Connected ANN.*** *Example of a simple fully connected neural network forming a Multi Layer Perceptron (MLP). The network can be seen to have three layers; one input layer with four input neurons, one fully connected hidden layer with five neurons and an output layer with a single output neuron.*

2.3.1 Convolutional Neural Architectures for Fine-Grained Recognition

Convolutional Neural Networks (abbreviated to CNN or ConvNet) are deep ANN architectures consisting of multiple convolutional, pooling and fully connected layers, typically used for processing imagery. Bearing resemblence with the ventral visual stream [181], CNNs draw inspiration from biology [207] in extracting a hierarchy of increasingly complex object feature representations [70]. Convolutional layers perform image convolution over tensors where kernel parameters are *learned* during training via the back-propagation algorithm [263]. This forms the crucial separation from classical, hand-crafted feature descriptors in computer vision (see Table 2.1 for examples). In the context of this work, the implica-

tion is that discriminative fine-grained visual features present amongst members of a population can be learned.

Algorithm	Paper Title	Citation
SIFT	"Object recognition from local scale-invariant features"	[199]
ASIFT	"ASIFT: A new framework for fully affine invariant image comparison"	[223]
PCA-SIFT	"PCA-SIFT: A more distinctive representation for local image descriptors"	[160]
ORB	"ORB: An efficient alternative to SIFT or SURF"	[268]
MSER	"Robust wide-baseline stereo from maximally stable extremal regions"	[206]
SURF	"SURF: Speeded up robust features"	[26]

Table 2.1: ***Hand-Crafted Feature Descriptors.*** *Examples of popular hand-crafted feature description algorithms prior to deep learning and learned convolutional feature representations.*

Seminal work in CNN architectures goes as far back as 1989 when back-propagation with a shallow CNN learned features to discriminate handwritten numbers/digits [185] from the famous MNIST dataset [184]. More recently in 2012, AlexNet [171] (pictured in Figure 2.5) proposed an eight-layer deep network and demonstrated state-of-the-art performance on ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 [269], containing a combination of general and fine-grained object categories (e.g. 'airliner' and 'space shuttle'). This new level of architectural complexity was made computationally-feasible by the utilisation of GPGPUs for expensive back-propagation, and the proposed network gave rise to new seminal literature and improvements in CNN research (e.g. VGGNet [289], ResNet [128], Xception [52], ZFNet [343]).

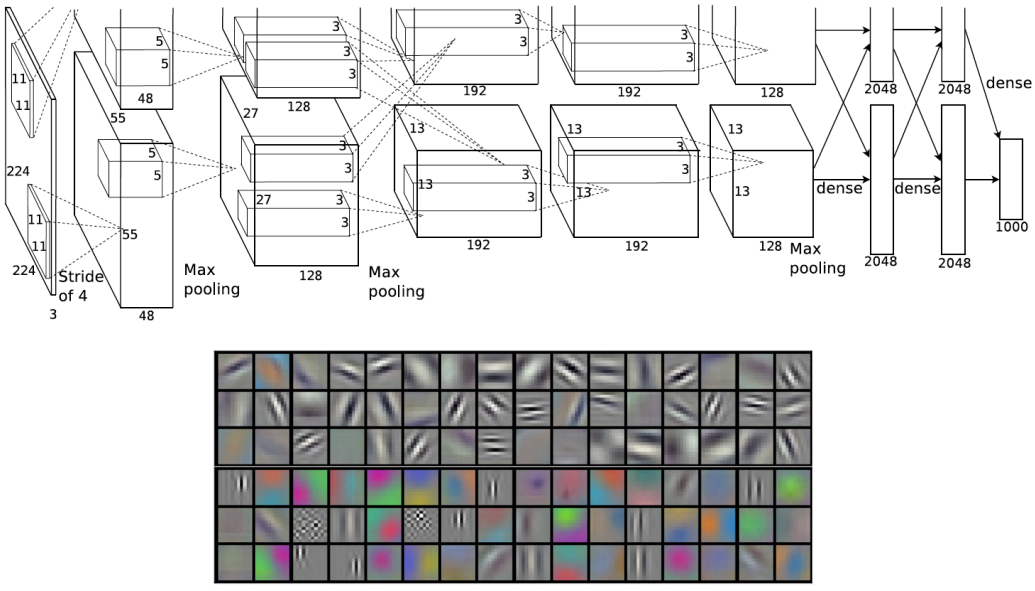


Figure 2.5: ***AlexNet CNN.*** *Figures from the seminal AlexNet CNN architecture [171] designed in 2012 to classify images for the ILSVRC 2010 contest [66]. (Top): the large and 8 layer-deep (for the time) ANN architecture comprised of five convolutional layers followed by three fully connected layers. Visible is the distribution of workload to two GPUs (top and bottom layers), with communication occuring at certain depths, illustrating the complexity of the architecture operating with limited GPGPU capabilities at the time. (Bottom): examples of learned* $11 \times 11 \times 3$ *convolutional filters by the first layer on both GPUs when trained for the ILSVRC. Image credit : Krizhevsky, A. et al. [171].*

In its first incarnation, the 2014 GoogLeNet CNN [302] set new state-of-the-art performance in the ILSVRC 2014 [270]. Success of the architecture lies in the design of "network in network" [195] Inception modules parallelising convolutions at multiple scales *with* dimensionality reduction result-

ing in many less required parameters. As such, the depth of the architecture can be increased whilst maintaining computational expense. Deep networks, however, then suffer from a diminishing ability to propagate gradients back through every layer due to the vanishing gradient problem [132, 134] – a problem also encountered when training recurrent networks [27, 239] (more on this in Section 2.3.3). This problem is solved by the addition of two auxiliary classifiers at intermediate depths such that, during back-propagation the gradient signal is amplified (see Figure 2.6). A subsequent variation on the architecture implementing batch normalisation [144] outperformed the best result on ImageNet at the time. Since then, new versions have introduced convolution factorisation (Inception v2) combined with applying batch normalisation on auxiliary classifier fully connected layers (Inception v2 + BN-auxiliary = Inception v3) [303] yielding substantial improvements over the start-of-the-art on ILSVRC 2012.

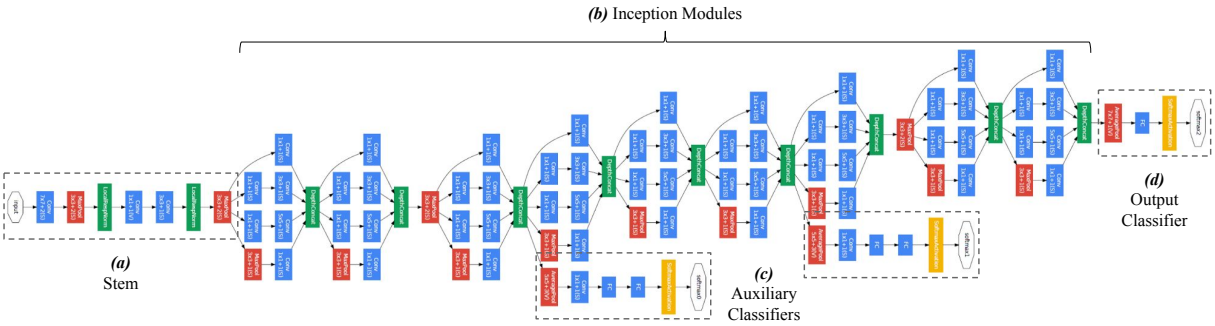


Figure 2.6: ***GoogLeNet CNN Architecture.*** *The wide and very deep GoogLeNet CNN architecture proposed in 2014 [302]. The pipeline consists of* ***(a)****: a small input CNN outputting into* ***(b)****: nine "network within a network" Inception modules with* ***(c)****: gradient signal-amplifying intermediary and auxiliary classifiers that, during training assist the* ***(d)****: final output classifier. Image credit : Szegedy, A. et al. [302].*

Due to the success of the various incarnations of Inception-based CNNs, especially on fine-grained categories present in ImageNet, the architecture is demonstrably well-aligned with the visual individual identification intentions of this thesis. This specifically corresponds to the ability to autonomously learn and detect particularly discriminative object features on a per-individual basis. Consequently, the architecture is used in this work to perform individual cattle identification in a standard feed-forward setting (see Section 7.6.1), but more commonly to extract image-wise feature vectors that are temporally integrated over multiple iterations or time-steps; a *process* (Chapters 4, 5 and 7). Methods performing temporal integration of viewpoints are explored further here in Section 2.3.3. Conversely, in settings where visual complexity of imagery is low, a simpler solution is preferable. Specific to the work in this thesis on automating agent-environment exploration decisions based on visual abstractions or simulated models (see Section 2.4 or Chapter 6), use of shallower networks is demonstrated to be sufficiently robust. In consideration of computational constraints imposed by processing on-board a UAV-based agent, this choice becomes important.

2.3.2 Object Detection via Regional Convolutional Neural Networks

Recent improvements in deep CNN architectures have seen them outperform traditional computer vision techniques in object detection [280, 109] as well as image classification tasks across benchmark datasets [171]. Regional Convolutional Neural Network (R-CNN) architectures combine these two tasks; candidate object locations are determined and subsequently classified [109]. Yet, in its original form, R-CNNs are computationally expensive to train and evaluate. With the introduction of sharing convolutions across proposals in Fast R-CNN [108] and SPPnet [127], significant improvements to efficiency were made, although region proposal computation remained the bottleneck. Circumventing this problem, Ren et al. propose the addition of a Region Proposal Network (RPN), which shares convolutional features with the detection network leading to Faster R-CNN [259] (see Figure 2.7 for an illustration).

Furthermore, Region-based Fully Convolutional Network (R-FCN) [61] – based on fully convolutional architectures, such as Fully Convolutional Network (FCN) [198] – go further and avoid the per-proposal evaluation of Fast and Faster R-CNN. This change results in a significant speedup over Faster R-CNN on the PASCAL Visual Object Classes (VOC) datasets [61].

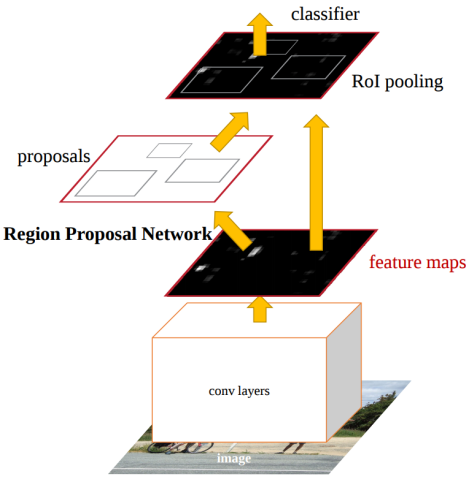


Figure 2.7: ***Faster R-CNN Architecture.*** *The Faster R-CNN architecture [259] improves model inference time of previous incarnations [109, 108] by proposing a new, convolutional feature-sharing Region Proposal Network (RPN) combined into a single, unified detection network. Image credit: Ren, S. et al. [259].*

At the time of writing, the current state-of-the-art in the task of "detect and classify" belongs to real-time detector You Only Look Once (YOLO)[1] [256]. The method casts frame detection as a regression problem by subdividing the image into $S \times S$-sized cells, each with associated bounding boxes (with scores) and class probabilities. In this form, the architecture can encode information about the entire image, as opposed to region proposal methods (e.g. Fast R-CNN [108]). A second, subsequent iteration yielded improved mean Average Precision (mAP) on benchmark datasets such as, PASCAL VOC [91] and Common Objects in Context (COCO) [196], whilst executing significantly faster [257] than other methods – an invaluable attribute when limited computational resources are available for real-time inference on-board a UAV agent. The most recent edition, YOLOv3 [258], proposes a marginally more complex architecture resulting in more accurate performance. Over the course of this thesis, the evolution of ANN-based detectors was witnessed such that, Chapter 3 employs the use of Faster R-CNN [259] (state-of-the-art at the time of work completion) and later chapters utilise the current-best YOLO detector [256, 257, 258] to capitalise on contemporary advancements (see Chapters 5 and 7).

In the context of this work, the detection and localisation of target objects (cattle) is frequently necessary. This is since, raw imagery from the agent is general and therefore, we need to determine which parts of the image (if any) contain objects of interest. In this work, detectors are trained on the Holstein Frieisan species generally, such that they can be located in relevant imagery. The product of their use yields sets of Region of Interest (RoI); target-centric sub-images that can then be provided as input into subsequent architectures estimating identity. In this form, the species detector focusses computational attention on a single individual's unique features by minimising the number of background pixels.

2.3.3 Recurrent Architectures for Temporal View Integration

Recurrent Neural Network (RNN) architectures introduce the notion of memory across multiple evaluations of a network by additionally outputting information to the subsequent iteration (as exemplified in Figure 2.8). This renders them useful for tasks such as image sequence captioning, where the processing of temporal or linked information requires retention, or, as is the case here, viewpoint integration over time. That is, retaining implicit model knowledge of object identity across multiple observations or images. This could be required in both passive iterative processes (the agent has no control over ob-

[1] YOLO website: https://pjreddie.com/darknet/yolo/

ject viewpoint; Chapter 4) and active iterative processes (the agent actively seeks particular viewpoints; Chapter 5).

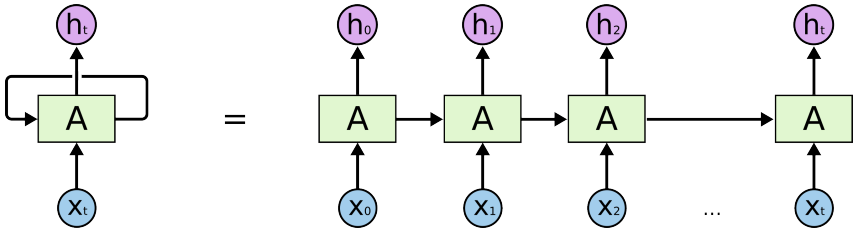


Figure 2.8: ***Unrolled RNN.*** *Temporally unrolled RNN for t time-steps, where data or memory is passed to the subsequent iteration. Image credit: Olah, C. [232].*

Despite their intuitive simplicity, basic RNNs are unable to learn long-term dependencies [27, 239] via traditional training methods such as Back-Propagation Through Time (BPTT) [327], Real-Time Recurrent Learning (RTRL) [261] or Truncated BPTT [299]. This is due to the temporal progression of back-propagated error values being exponentially dependent on the size of weights [133]. First introduced in 1997, Long-Short Term Memory (LSTM) units [135] eradicate this problem by design and are illustrated in Figure 2.9. The architecture of LSTM cells or units enforce constant error flow, therefore preventing back-propagated errors from exploding or vanishing [133].

Following the success of LSTM networks, extensions or variations upon the original cell architecture have included allowing gate layers to examine the cell state (Peephole LSTM) [104], arranging networks of LSTM cells in a multidimensional grid [159], introducing a depth gate to connect memory cells of adjacent layers (Depth-Gated LSTM) [336], and many more. Proposed in 2014, Gated Recurrent Unit (GRU) [50] have been gaining popularity. They combine LSTM forget and input gates together into a single "update" gate as well as other modifications resulting in a simpler model overall with fewer parameters than LSTM cells. However, despite all these efforts, Greff et al. [115] find that no variants significantly outperform the original LSTM architecture over large-scale analysis of eight LSTM variants covering three popular machine learning tasks. Furthermore, Jozefowicz et al. [157] identify a RNN architecture that outperforms both LSTM and GRU units at only a small proportion of evaluated tasks. Consequently, standard LSTM layers are utilised throughout this thesis where recurrency is required for temporal integration of convolutional features from multiple object observations.

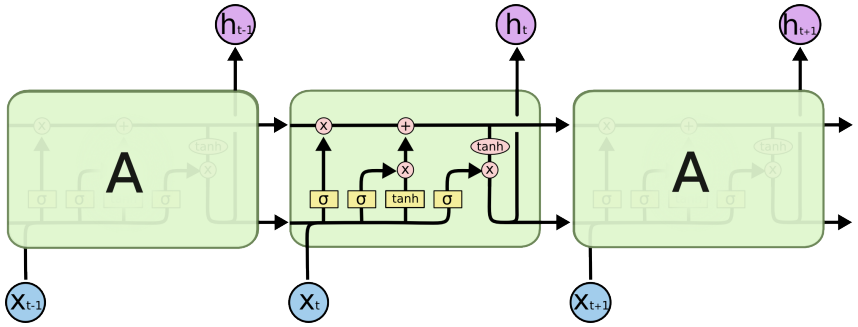


Figure 2.9: ***Unrolled LSTM-based RNN.*** *Temporally unrolled RNN for three time-steps with the recurrent layer as a LSTM unit. Visible are the input X_t and output h_t components at each time-step. Internally, a first sigmoid layer chooses to allow or disallow the output of the previous timestep h_{t-1} concatenated with the current input X_t to update the cell state, called the the forget gate. Next, consideration of new information additively affecting the cell state is assessed by a sigmoid input gate i_t combined with hyperbolic tangent candidate values $\tilde{C}_t$ on X_t, h_{t-1}. Subsequently, the cell state is updated C_t based on previous steps. To finalise, another sigmoid layer combined with another hyperbolic tangent gate dictate LSTM layer output for this time-step h_t. The output h_t, cell state C_t and subsequent input X_{t+1} form inputs into the proceeding time-step, completing the pipeline. Image credit: Olah, C. [232].*

2.3.4 Circumventing Deep Reinforcement Learning for Active Object Recognition

Seminal work in Deep Reinforcement Learning (DRL) occurred in 2013 when researchers from DeepMind Technologies used a Deep Neural Network (DNN) as a function approximator for learning control policies from high-dimensional sensory input such as images. Trained with a variant of Q-learning towards the goal of replicating human-level performance in playing Atari 2600 games, the term Deep Q-Network (DQN) was coined [219, 220]. DRQNs [126] extend this work in a recurrent design for partial agent-environment observability with the utilisation of a LSTM layer [135] in place of the first post-convolutional fully connected layer (see Figure 2.10). These works gave rise to a new genre of reinforcement learning research [218, 193, 315, 322], where DRL has demonstrated incredible success in a broad range of applications [224, 187].

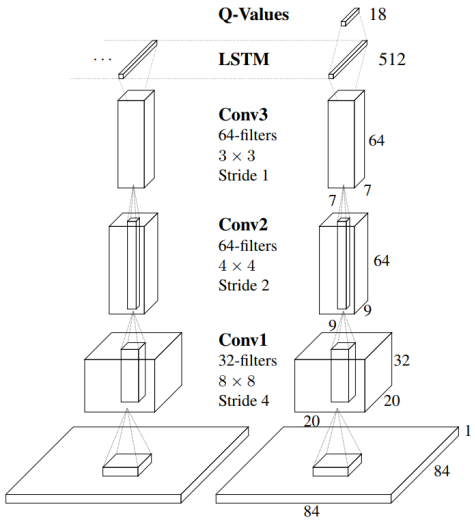


Figure 2.10: ***Deep Recurrent Q-Network (DRQN) Architecture.*** *The recurrent DRL architecture [126] over two time-steps forming a Long-term Recrrent Convolutional Network (LRCN); an ANN comprised of convolutional layers extracting spatial features temporally integrated by a subsequent LSTM layer. LSTM outputs form Q-Values – estimates of long-term expected reward from executing an action in a given state [325] – following a final fully connected layer. Image credit: Hausknecht, M. et al. [126].*

(Deep) reinforcement learning has even played a part in the task of active object recognition [235]; jointly predicting object labels visually whilst selecting subsequent actions attempting to improve recognition performance – aligning strongly with the work in this thesis towards active involvement of a freely moving agent identifying a target subject (as will be explored fully in Chapter 5). One such work involves use of the GERMS 2015 dataset[2] [202], featuring 136 object categories of stuffed toys representing a variety of micro-organisms, with limited intra-population visual similarities. For each category, the dataset consists of robot-centric imagery whilst grasping the object amid 1-DoF through-range wrist rotation. In this form, researchers proposing new algorithms can simulate robot wrist joint values towards validating their active recognition approach. Included in the original paper [202] is a benchmark solution founded in Deep Q-Learning, that was later extended by the same authors [201] to train a full architecture end-to-end.

In regards to the work in this thesis on active identification, the agent (a quadrotor UAV) is modelled to have significantly more DoF – such as 5-DoF in Chapter 5. The resulting objectivity landscape constitutes higher dimensionality and searching this space becomes costly, especially in reality. Instead here, an abstract formulation of discrete, finite viewpoint sequences are exhaustively searched for per-individual least-costly satisfactory solutions (see Chapter 5 for further details). Whilst (deep) reinforcement learning is proven to be well suited to one-dimensional action selection [202, 201], this thesis proposes its application to be unnecessarily complex when the finite action set has low cardinality. In other words, if detailed training information is easily available, then reward-based learning is not required and the state space to be learned can be explored more efficiently and effectively due to known best solutions. This intuition is demonstrated in this thesis, with Chapter 5 exhaustively searching abstracted viewpoints, and Chapter 6 learning from examples of optimal navigation decisions.

[2]GERMS dataset website: `http://rubi.ucsd.edu/GERMS/`

2.4 Environment Exploration

Thus far, methods have been described that identify content based on object presence in a frame or sequence of frames. Next, methods of active navigation will be reviewed that allow an agent to get into a position whereby target objects of interest are within the frame to begin with. In other words, reviewing exploratory strategies needed to discover new individuals in an environment. The search for targets with unknown locations typically arises in Search and Rescue (SAR)-based applications [194, 319, 271].

2.4.1 Classification for Navigation

Robotic navigational research naturally relies heavily on the categorisation of sensory input. With respect to visually-motivated navigation, approaches can largely be categorised into map-based and mapless navigation [34, 68]. Mapless vision approaches – with no global environment representation – can be found to: be based in optical-flow [274] and template appearance matching [209, 155], landmark feature tracking [241, 272] and more recently relevant, directly classify visual input via CNNs [110, 254]. As a particular example, Pomerleau, A. D. seminally employed an ANN to drive an offroad vehicle with input being a 30×32 visual field in 1991 [245]. Neuron weights were learnt/trained via back-propagation on expert (human) steering reactions whilst forward propagation yields image categorisation into 30 classes for vehicle steering (e.g. $\{$sharp left, ..., straight ahead, ..., sharp right$\}$). In this thesis however, a two-dimensional global approximation of the environment (storing visited positions) inspired by occupancy grid maps [35, 36, 234] is formulated, as opposed to post-exploration map-building approaches [222] and topological map representations [167].

2.4.2 Deep Reinforcement Learning for Exploration

With respect to the portion of the work in this thesis involving agent-environment exploration attempting to learn efficient, domain-specific navigation policies (i.e. a UAV-based agent efficiently discovering the positions of cows in a field), a proportion of current research resides in DRL-based approaches. A general literature review on deep reinforcement learning is given in previous Section 2.3.4. Literature relevant to exploration however, has involved learning agent navigation in complex [216] and similar [344] environments, map-less navigation [305] achieved visually [121] or via other sensor measurements. One particularly noteworthy paper employs DRL towards target-driven visual agent navigation in simulated indoor environments [360] – bearing resemblance with the problem formulation of this work solving exploratory agency (specifically, Chapter 6). As argued properly in Section 2.3.4, in this thesis, DRL is not used towards environment exploration because knowledge of what constitutes a good solution (e.g. via Travelling Salesman Problem solutions) is known, and can be trained against.

2.5 Visual Animal Biometrics

Evolution adapts species to their environments over many generations. In the case of animals in the wild, exhibited *coat patterns* have become increasingly tailored towards their surroundings to provide camouflage or disruptive colouration [60] – examples for various species are given in Figure 2.11. Concerning visual animal biometrics, there is therefore, an intrinsic challenge in automatically detecting animals exhibiting such coat patterns in relevant imagery. Whilst principally maintaining the visual theme amongst a species, small perturbations during the formation of coat patterns yield individuality. The result of which is visible in the form of individually-unique scapular stripes for zebras, dorsal spot arrangements for manta rays and coat pattern structures, markings and alignments for Holstein Friesian

cattle. This characteristic is exploited in this thesis, building upon established methodologies for various species [44, 172, 175]. Applications of automating such processes typically reside in conservation work [273, 178, 107]. The following Section 2.6.1 discusses one area of conservation research dedicated to the involvement of flying robots, aligning with this work.

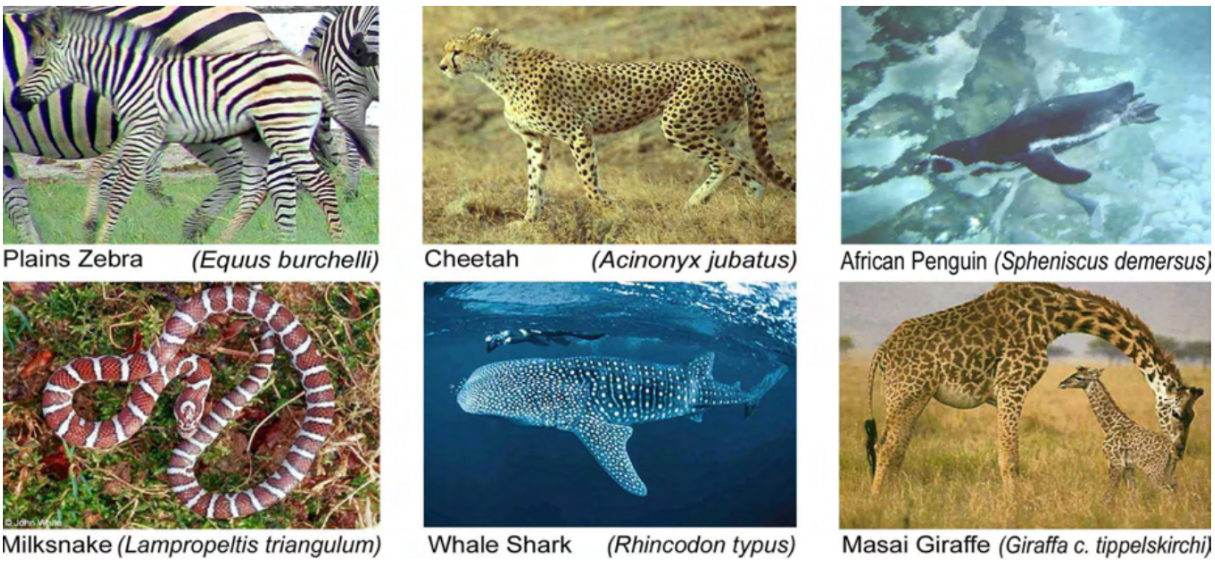


Figure 2.11: ***Species with Coat Patterns.*** *Examples of several species exhibiting variant coat patterns that have evolved to their habitat/surroundings in order to provide camouflage or disruptive colouration [60] – as is visible for Plains Zebra. Figure credit: Burghardt, T. [44].*

2.6 Unmanned Aerial Vehicles

Relatively low-cost Unmanned Aerial Vehicle (UAV) are becoming evermore popular tools for the realisation of research interests [192, 55], commercial purposes (e.g. power line inspection [65, 153, 190], wind farm inspection [226, 298]) as well as for consumer aerial photography and videography. With respect to academia alone, research-impeding factors such as safety, cost, logistics and experiment repeatability that are present in manned aerial missions are certainly now less of a problem.

2.6.1 Application in Conservation

One area of applied research where UAVs have made a distinct impact is conservation, which aligns with the social monitoring interests of this thesis. This follows from wildlife management and conservation missions being adequately solved by unmanned flight systems with medium to low weight/size payload capabilities. Put differently, manned light aircraft or rotorcraft are usually an overkill. Multiple existing works assess and ultimately commend the effectiveness of fixed-wing UAVs (see Figure 2.12b) for aerial surveillance of wildlife over large distances or areas, recommending flight system-specific details for optimal flights (e.g. electrically powered, hand-launchable) [156, 154]. Boasting long flight times and possible distance ranges out of single charges/flights due to gliding and efficiency capabilities, fixed-wing UAV or Unmanned Aircraft System (UAS) platforms are suitable for missions where this is pertinent. The ability for multi-rotor (typically quad-rotor) UAVs (see Figure 2.12a) to accurately hover or hold position and orientation, however, is a definite requirement for the monitoring goals of this research. Combined with multi-rotor UAV capabilities for Vertical TakeOff and Landing (VTOL), their wide availability, low-cost, ease of use and integration with existing software packages as well as many other factors, they are the clear choice of autonomous aerial vehicle form factor for the purpose of this project.

(a) Quadrotor UAV – The DJI Matrice 100; the flight platform used as part of this thesis.

(b) Fixed-wing unmanned aircraft system. Image credit: A. Gay, et al. [102].

*Figure 2.12: **UAV Platforms.** Example form factors for commonly-utilised UAV in conservation, research and consumer sectors*

Significant work towards the monitoring or conservation of animals, wildlife management and nature utilising differing UAV systems and setups has taken place. Looking towards the growing problem of global deforestation, costly solutions of effective data gathering have typically been employed (e.g. satellite imagery, airborne sensors, manned flights). Koh et al. [165] propose a low-cost (sub $2000), autonomous UAV flight system which captures high resolution aerial imagery of target areas automatically. The implication of inexpensiveness being that previously inaccessible data can now be captured by researchers in developing countries, low budget projects, etc. Non-technical operators can easily create a flight mission by simply defining GPS waypoints – similarly to the commercial product DroneDeploy [77]; a user-friendly mobile and browser-based application for mapping target areas. In fact, this use of UAVs for high resolution aerial imagery capture for conservation pursuits is common in research for aforementioned reasons [106, 105, 309].

UAVs are also commonly being used in the area of conservation research or monitoring of wildlife [136, 168, 1, 262] by counting individuals, capturing high-resolution imagery of animals and more. Van Gemert, J. et al. use a quadrotor drone and propose a system for forming estimates of animal populations in particular areas [314]. The authors propose that this could be achieved automatically on-board the flight system. They compare solutions between the use of a R-CNN or, methods based on bag-of-words for automated object detection. Similarly in application, Israel, M. describes a UAV system for the detection of roe deer fawn and other animals [146]. The animals are often found sleeping or resting in farm areas where they could be killed by agricultural machinery. The system uses an on-board, lightweight thermal infra-red camera for animal detection which flags GPS coordinates to a human operator for animal capture and release in to the wild. In a fashion similar to DroneDeploy [77], users manually define a scanning region and a flight path is automatically generated and executed by the UAV. Very similarly, Christiansen, P. et al. [54] employ the use of thermal imaging for automated animal detection and classification to promote wildlife-friendly farming by detecting animals within user-defined areas due to be harvested/mowed. As demonstrated, UAV systems have become popular for wildlife and livestock monitoring in research [30, 248, 102] as a low-cost and effective solution.

In this work, a low-cost quadrotor UAV flight platform is used to ascertain the identities of individual Holstein Friesian cattle using an on-board RGB camera. The important feat being that area surveillance is performed with online and active agency towards efficient target discovery and robust identity estimation,

instead of passively analysing obtained imagery/footage offline. Consequently, the flight platform – the DJI M100 (depicted in Figure 2.12a) – was modified to house computational processing on-board (see Section 7.2 for further hardware details).

2.6.2 Aircraft Principal Axes and Control

In the utilisation of aerial platforms, it is necessary to define consistent reference frames alongside appropriate aircraft control algorithms, as this section will explore. The aircraft coordinate frame utilised throughout this thesis is illustrated in Figure 2.13 and is outlined as follows. Specifically, ϕ: roll, θ: pitch, ψ: yaw RPY angles operate counter-clockwise about x, y, z denoting the longitudinal, lateral and vertical axes, respectively. Additionally, $+x$ is aligned with the front of the aircraft, $+y$ with the left hand side and $+z$ is in the upwards direction globally; forming a right-handed Front Left Up (FLU) aircraft reference frame. The most commonly used alternative specification is aircraft Front Right Down (FRD) to be aligned with North East Down (NED) geodetically.

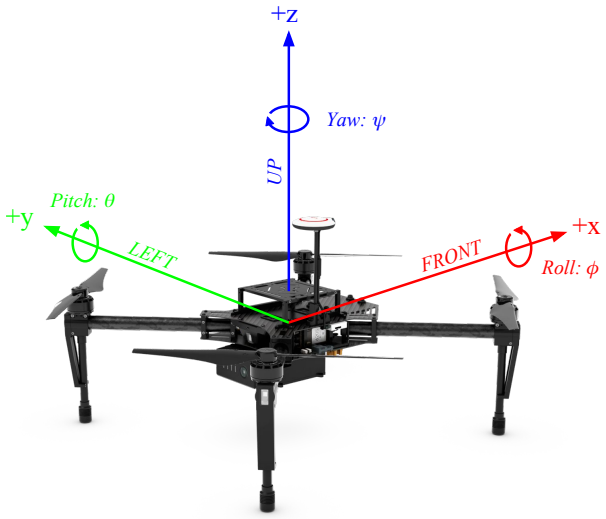


*Figure 2.13: **UAV Coordinate Frame.** Illustrated UAV Front Left Up (FLU) coordinate frame commonly used in aerospace engineering with corresponding roll, pitch, yaw (RPY) angles denoted by ϕ, θ, ψ, respectively. The front of the aircraft is aligned with the $+x$ axis and positive rotations are performed counter-clockwise.*[3]

The control of UAV position, orientation (attitude) and more is implemented via Proportional Integral Derivative (PID) control. Throughout this thesis however, low-level dealings are abstracted away from programmers of the employed UAV flight platform; control is implemented internally via PID. The choice of UAV is the DJI Matrice 100, and full hardware considerations are given in Section 7.2. It is however useful to understand how such control operates in order to reasonably diagnose exhibited flight behaviours. PID controllers are implemented on-board the UAV for controlling its position and orientation. When the user requests a goal state, a PID controller for each aircraft DoF (yielding six independent variables: $x, y, z, \phi, \theta, \psi$) smoothly realises that position/orientation. PID control for a single variable is described in the following paragraph and is depicted in Figure 2.14.

For a goal value $r(t)$ (the setpoint) at time t, the current error $e(t)$ is calculated from the difference between the setpoint and the current measurement of the variable $y(t)$ (process variable). From the error term $e(t)$, proportional, integrative and derivative control tuned by scalar gain values K_p, K_i, K_d, respectively is summated to yield a control adjustment $u(t)$ that is performed. From there, a new measurement is taken of the process variable $y(t)$, completing the cycle. Then typically, when the new measurement $y(t)$ is sufficiently similar to the setpoint $r(t)$ (the error $e(t)$ is below some threshold), this will be indicated to a higher monitoring process. Note that real implementations often selectively use individual components for domain-specific purposes (e.g. PI, PD, P), as is the case in Section 5.4.1 implementing a proportional controller for camera angle control.

[3] UAV image credit: DJI

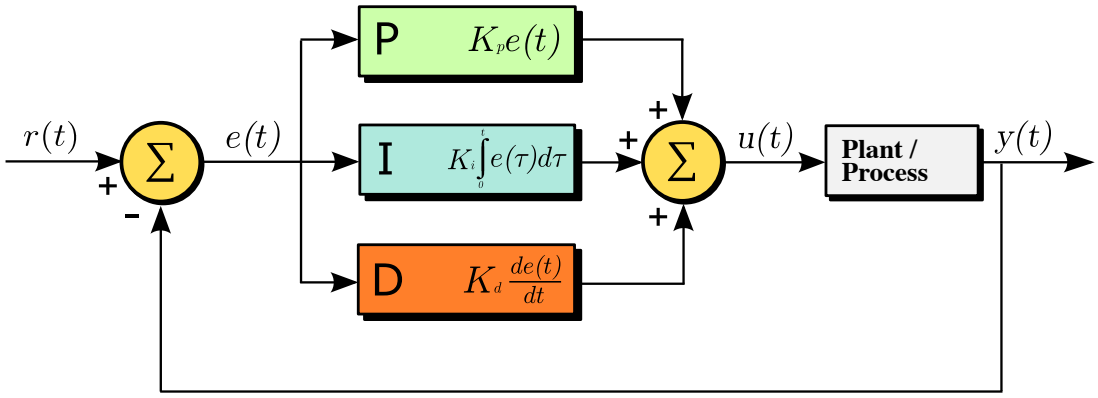


Figure 2.14: ***Proportional Integral Derivative (PID) Controller.*** *Diagram illustration*[4] *of an iterative PID controller attempting to control a measured variable $y(t)$ towards a goal value; the setpoint $r(t)$.*

2.6.3 Geodesy

Geodesy, also known as *geodetics*, refers to the scientific field of understanding the intrinsic properties of the earth's shape geometrically, its gravitational field as well as its orientation in space [228]. In the context of this thesis, established geodetic systems are used to discuss agent/UAV positioning in relevant reference frames alongside transformations between those reference frames, as required by the Application Programming Interface (API)s used in this work.

To begin, the most commonly known positioning system is GPS [249], a radio navigation system comprised of 31 satellites in sub-synchronous earth orbit owned by the United States government. Devices with GPS capabilities can localise to metre-level accuracy given line of sight to ≥ 4 satellites. Resulting localisations arrive in the form of the World Geodetic System (WGS)-84 [63] coordinates (a commonly-used Geographical Coordinate System (GCS) standard) to define any point on the earth:

$$(\varphi : latitude,\ \lambda : longitude,\ h : altitude), \tag{2.2}$$

where $\varphi \in [0,90]°N$ *or* $[0,90]°S$, $\lambda \in [0,180]°E$ *or* $[0,180]°W$, $h \in \mathbb{R}^+$m. This spherical coordinate system is fixed to the earth (hence being indepedent of the earth's on-axis rotation) such that $\varphi = 0°$ intersects the equator and $\lambda = 0°$ intersects the Prime Meridian.

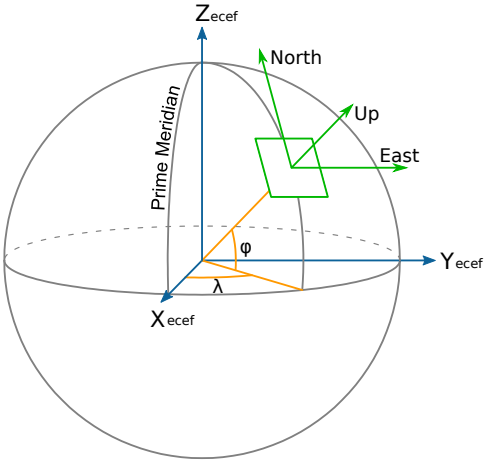


Figure 2.15: ***Earth Geographical Coordinate Systems.*** *Illustration*[5] *of the earth and relevant Geographical Coordinate System (GCS). The ECEF originates from the earth's centre of mass with orthogonal axes intersecting true north and the intersection point between the equator with the Prime Meridian. Spherical GPS coordinates (λ, φ) allow local reference frames to be defined at some length h: altitude, typically aligned with East North Up (ENU) or North East Down (NED).*

Next is the earth's Cartesian reference frame: Earth-Centered Earth-Fixed (ECEF), which is also indepedent of the earth's rotation, with the point $(0,0,0)$ defining the earth's centre of mass [56]. The x-axis

[4] Diagram credit: Arturo Urquizo – https://commons.wikimedia.org/wiki/File:PID.svg

[5] Diagram credit: Mike1024 – https://en.wikipedia.org/wiki/File:ECEF_ENU_Longitude_Latitude_relationships.svg

of ECEF is aligned with latitudinal and longitudinal coordinates 0° and 0°, respectively, and z is aligned with true north. To complete the definition, the y-axis also has latitudinal value 0° but is orthogonal to both x, y in a right-handed coordinate system.

It is often desirable to describe agents resolved globally in either of these GCS in a local Cartesian frame. This manually placed frame is usually aligned with the earth's axes in order to maintain operational simplicity; typically, East North Up (ENU) or North East Down (NED) for the x, y, z axes, respectively. As such, transformations are required when interchanging employed GCS to describe the agent's position with respect to the earth. In this work, WGS-84 coordinates are often transformed into a local Cartesian frame via ECEF; this mathematical process is described fully in Appendix B.

2.7 Multi-Object Tracking

Multi-Object Tracking (MOT) refers to the task of predicting the trajectories of a set of objects over the course of a sequence of images. Initial object regions are typically tracked via the paradigm of "tracking-by-(re)detection" [291, 334], where a popular approach involves a classifier trained online testing candidate regions [346, 158, 20] amongst alternatives [29, 41, 89, 278]. The challenge in associating objects across constituent frames resides in object autonomy, camera motion and the birth and death of trajectories; object instances entering and leaving the frame. In relation to the work at hand, use of "tracking-by-detection" is employed to track individuals in aerial image sequences, where initial object regions are inferred from object detectors. Tracklets (object regions over time) are then given to components inferring identity iteratively; over multiple variant observations. Specifically, Chapter 4 on passive iterative identification associates candidate trajectories via the Kernelised Correlation Filter (KCF) algorithm [131], whilst later Chapters 5 and 7 capitalise on real-time detection advancements using YOLO [256, 257, 258] in conjunction with simple association heuristics.

2.8 Cattle Identification Methods

The problem of automated biometric bovine identification has been well-studied over associated literature. Automated approaches can largely be separated into three categories [17]: *(1)*: those that utilise cattle muzzle patterns, *(2)*: rarer systems that employ retinal biometrics [10], facial features [180, 45] (equally applicable to pigs [125]) or body scans [14], *or* as is the case here, *(3)*: those that exploit coat pattern characteristics [204]. Note that advanced non-biometric computerised vision schemes for identification exist too, including work that takes advantage of the European cattle ear-tagging mandate [237] by applying text and character recognition in order to match tagged individuals against respective Bovine Identification Documents [317].

2.8.1 Muzzle Patterns

Bovine muzzle patterns were first introduced as an individually-unique, dermatoglyphic trait [217] as far back as 1922 [242]. This seminal work was extended in 1993 where significant differences in cattle dermatoglyphics across breeds were discovered [23]. Muzzle patterns form the foundation of significant research into semi-automated bovine identification methods [176, 179, 162, 308]. In many cases, standard feature-description algorithms such as Scale-Invariant Feature Transform (SIFT) [199] or Speeded Up Robust Features (SURF) [26] are employed [8]. In particular, Noviyanto et al. utilise SIFT on 160 manually-acquired muzzle prints of 20 individuals after applying ink to the cattle's muzzle [230], similarly to Minagawa et al. [215].

These semi-automated solutions are operationally improved by approaches operating on muzzle images alone [177, 229, 82, 24, 342, 85, 99]. As an example, Awad et al. combine SIFT and Random Sample Consensus (RANSAC) for feature extraction and filtering to achieve strong identification accuracy on muzzle images [19, 18]. More recently, robust muzzle differentiation has been achieved by methods involving ANNs on hand-crafted features [84, 83] as well as basic, shallow convolutional architectures [173].

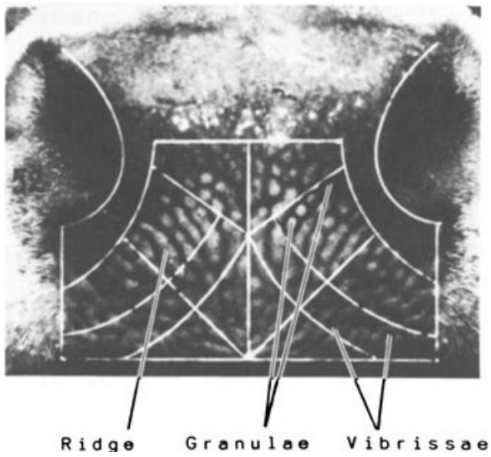


Figure 2.16: ***Cattle Muzzle/Nose.*** *Example of the cattle muzzle used as a dermatoglyphic biometric trait [242]. Identification is performed by analysing spatial ridge features distributionspermitted by taking muzzle/nose print images using ink imprints. Image credit: Baranov, A. S. et al. [23].*

To reiterate, whilst effective use of muzzle patterns as a biometric entity is provenly robust across differing species [23] in aforementioned literature, acquisition of satisfactory imagery is inherently cumbersome and difficult. To begin with, muzzle prints were painstakingly and intrusively manually acquired by transferring applied ink from the cow's muzzle to paper for subsequent analysis. This is no longer the case given contemporary image analysis algorithms (e.g. [174, 306, 200, 86, 307, 81]) operating on muzzle images. However acquisition of such images is difficult and requires heavy preprocessing, given the small size of the region (and the corresponding minimum resolution of features), viewpoint variability from animal autonomy, illumination difficulties from moisture variation and lighting change (especially since muzzle regions are particularly monotonal). Further still, reliably producing muzzle images automatically must pass a robust detection stage. There is therefore, a justified gap in literature for *easily-automated and minimally-invasive identification algorithms*. A gap this thesis attempts to address.

2.8.2 Coat-Pattern Analysis

To re-iterate, Holstein Friesian cattle exhibit individually-distinctive black and white (sometimes brown and white) patterns and markings over their bodies due to piebald[6] spotting [236, 75]. Dorsal patterns alone, as exemplified in Figure 2.17, form complex visual alignments and structures. The application goal of this thesis is, therefore, to exploit visual uniqueness exhibited in dorsal markings to perform automated and minimally-intrusive individual identification. Acquisition of appropriate imagery is relatively straightforward; aerial images can be captured by downward-facing cameras in barn or with low-cost UAV-based agents outdoors – as is tested in both cases in this thesis. The obvious downfalls to this biometric assumption is that the proposed methods are inherently limited to (a): species exhibiting such coat patterns to begin with, and (b): individuals of a population that exhibit sufficient pattern variation – a general requirement in biometrics [44]. As a simple example, two entirely black individuals are impossible to differentiate under this assumption, and alternative biometric methods would have to be sought.

First attempts to utilise Holstein Friesian coat patterns for identification are very limited. Martinez-Ortiz, C. et al. [204] propose use of Principal Component Analysis (PCA) to infer nearest neighbour identity

[6] 'Piebald' Oxford dictionary definition: having irregular patches of two colours, typically black and white.

Figure 2.17: ***Holstein Friesian Cattle.*** *Example image of Holstein Friesian cattle grazing freely at the University of Bristol's Wyndhurst veterinarian farm in Langford Village, UK. Imagery was acquired aerially via the use of a DJI Inspire MkI UAV utilised later in this thesis for offline model training and validation (see Chapters 3, 4). With respect to original contribution in the application field, the goal of this thesis is to exploit visible per-individual coat pattern uniqueness for minimally-intrusive visual identification purposes.*

from known high-dimensionality templates. The authors additionally propose use of a scale invariant feature, specifically SIFT [199], to locally describe exhibited pattern characteristics. The core logic being that when two images containing the same individual are compared, there will likely be more feature similarities (matches) than when two different individuals are examined. This consequently formed inspiration for the first part of this thesis describing individual identification via matching local coat pattern features (see Section 3.2). The novelty in the methodology employed here, however, lies within the realisation that a proportion of extracted features will not contribute beneficially to identification (as is visible in Figure 2.18). As such, can the properties and attributes of such features be effectively learnt and ultimately, discriminated during evaluation (see Section 3.2.2)?

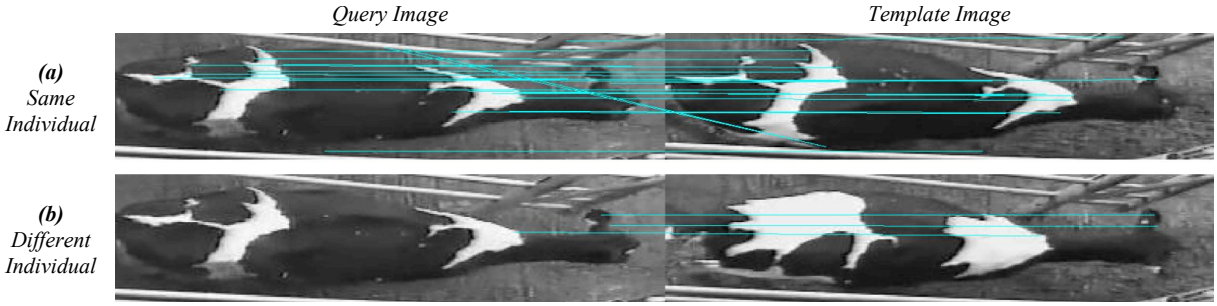


Figure 2.18: ***Pairwise Local Feature Matching.*** *Individual identification of cattle from pairwise image comparisons. Features are extracted on a query image and compared against those from a known set of templates. Image pairs containing* ***(a)****: the same individual will likely contain more feature similarities according to some matching metric compared to* ***(b)****: pairs with differing individuals. Image credit: Martinez-Ortiz, C. et al. [204] .*

Additional identification difficulties arise in unconstrained imagery from varying viewpoints and the non-rigidity of animals, that is, skin deformation due to changes in pose and articulation. It is for this

reason that this thesis employs the use of fully affine-invariant feature extraction algorithm Affine Scale-Invariant Feature Transform (ASIFT) [223, 339, 340] to explicitly model affine deformations of the cattle coat (see Section 3.2). This thereby increases the number and quality of the matchable baselines compared to alternative local descriptor techniques such as SIFT [199]. This traditional feature-based approach is built upon by directly applying convolutional ANN architectures to learn individually-characteristic features automatically in a supervised manner (see Section 3.3). The core takeaway being that, within the scope of the experiments conducted in this thesis, (a): individually-exhibited dorsal coat patterns are sufficiently distinctive across experimented herd sizes (e.g. ≤ 100) *and*, (b): the methodology and pipelines proposed here can reliably infer identity from such patterns. Importantly, and with respect to the application of individual cattle identification towards agricultural automation, this can occur with little disturbance to the animals.

2.9 Summary

To summarise the topics presented within this chapter (in order) alongside their relationship to this work, the following table is given with reference to relevant parts in the work chapters of this thesis:

General Topic / *Sub Topic*	*Review Section*	*Specific Work Topic*	*Specific Work Chapter/Section(s)*
Active Object Recognition	2.2	Active Iterative Individual Identification	5
Next Best View	2.2.1	Viewpoint Sequence Generation	5.4.5
Object Pose Estimation	2.2.2	Agent-Target Displacement Estimation	5.4.2
Artificial Neural Networks	2.3	-	-
CNNs for Fine-Grained Recognition	2.3.1	Identity Classification / Feature Extraction	3.3 / 4.4, 5.4.4, 7.4.3
Object Detection via R-CNN	2.3.2	Identity Classification / Species Detection	3.3.1 / 4.3
Recurrent Architectures	2.3.3	Temporal Viewpoint Integration	4, 4.4, 5.4.4, 7.4.3
DRL for Active Object Recognition	2.3.4	-	-
Environment Exploration	2.4	Exploratory Agency	6
Classification for Navigation	2.4.1	Exploration Problem Discretisation	6.3
DRL for Exploration	2.4.2	-	-
Visual Animal Biometrics	2.5	-	-
Unmanned Aerial Vehicles	2.6	Data Acquisition / Real-World Experimentation	4.2, 7.3 / 7
Application in Conservation	2.6.1	Cattle Census	7.6
Principal Axes and Control	2.6.2	UAV Setup and Control	7.5
Geodesy	2.6.3	GPS Coordinate Fulfilment	7.5.3
Multi-Object Tracking	2.7	Individual Region Tracking over Frames	4.4, 5.4.1, 7.4.1
Cattle Identification Methods	2.8	-	-
Muzzle Patterns	2.8.1	-	-
Coat-Pattern Analysis	2.8.2	Local Feature Matching	3.2

Chapter 3

Single Frame Identification

3.1 Chapter Overview

This chapter primarily aims at demonstrating that automated visual identification of Holstein Friesian cattle can occur effectively, to a certain extent, through the traditional computer vision paradigm of evaluating a *single* image. Holstein Freisians exhibit coat patterns that are exploited in this thesis for individual discrimination. The difficulty lies in intra-population visual similarities, a factor which increases proportionally with the size of the population, rendering this task an example of fine-grained identification.

Pitfalls to the concept of single frame identification are experimentally highlighted by individual- and circumstantial-specific causes. These issues are addressed by later chapters whereby multiple, varying images of an individual are analysed in a passive and active setting. However, within the scope of this chapter, an effective demonstration is given for single-frame identification across two approaches:

1. **Local Feature Matching (Section 3.2)**: utilising a standard feature descriptor in computer vision (ASIFT [223]) for matching local features in preprocessed RGB-D imagery towards effectively recovering individual identities. Deployment of this approach however, comes at significant computational cost.
2. **Deep Learning (Section 3.3)**: employing a R-CNN-based architecture for individual cattle identification operating on top-down imagery acquired in a real-world agricultural environment, similarly to the section above. In doing so, convolutional architectures are validated on the task of selecting, learning and discriminating features that contribute towards individuality for each member of a population.

The key takeaway is that the utilisation of dorsal features alone as a biometric entity is demonstrably viable across the population sizes used in this chapter ($\leq$100 individuals). The consequence of which is significant; the automatic and minimally-intrusive identification of Holstein Friesian cattle is eminently possible.

3.2 Local Feature Matching

The objective of this section is to provide a baseline for identification using traditional techniques. This is achieved by automating the visual identification of individual Holstein Friesian cattle from dorsal RGB-D imagery taken in a real indoor farm environment using classical, hand-crafted features. This section contributes a dataset and proposes a system that can reliably derive animal identities from top-down stills by firstly, depth-segmenting animals in RGB-D frames, and then extracting a subset of local ASIFT coat descriptors predicted as sufficiently individually distinctive across the species. Predictions are generated by a Support Vector Machine (SVM) using Radial Basis Function (RBF) kernels for predictions based on the ASIFT descriptor structure. Learning such a species-specific identity model is shown here to be effective, and robustness to poor or complex input image conditions such as the presence of multiple cattle, bad depth segmentation, etc. is demonstrated. The proposed system yields 96.6% identification accuracy over a testing dataset covering a herd of 25 individuals from the published FresianCattle2015 Dataset.

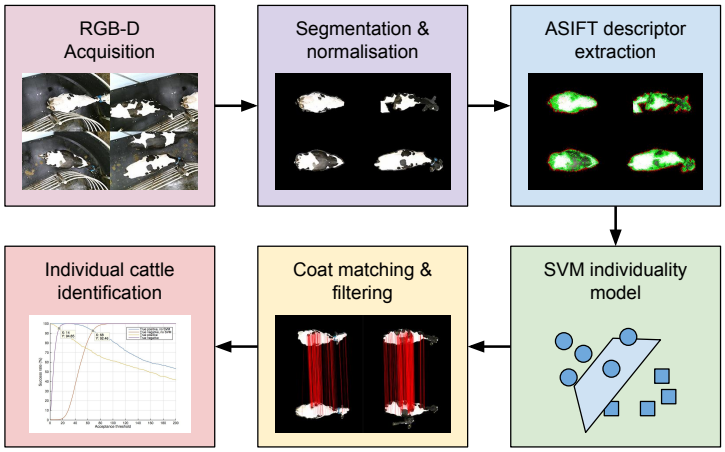


Figure 3.1: ***Proposed Cattle Identification Approach.*** *The approach taken in this chapter segments animal regions via fitting a depth model, and then extracts ASIFT descriptors over the detected area. A SVM is used to learn a species-wide predictor of descriptor-individuality employed to select and use features for cattle identity recovery.*

3.2.1 Dataset

Data Acquisition

Data acquisition was implemented at the Wyndhurst Farm at Langford Village, UK filming cattle exiting the milking file and freely walking towards holding pens[1]. As can be seen in Figure 3.2, the passage-way is sufficiently narrow in order to ensure a one-way system (it would be difficult for a cow to turn around in that space), which later aids the rotation-normalisation phase in image preprocessing (since all cows should face the same direction; facing right). A top-down operating Kinect 2 sensor was used in conjunction with a dedicated workstation for aligned RGB-D recording. Data was captured from this overhead perspective at approximately 4m above the ground, where, 16-bit depth imagery at a resolution of 512×424 and raw RGB video at 1920×1080 at 30fps (full high definition) was recorded.

Image Preprocessing

Cattle in the raw image frame (as exemplified in Figure 3.2) are first segmented against the background and normalised for rotation. Towards this goal, associated depth maps were thresholded at empirically

[1] Many thanks to Dr Sion Hannuna and Dr Neill Campbell for acquiring, labelling and preprocessing this dataset.

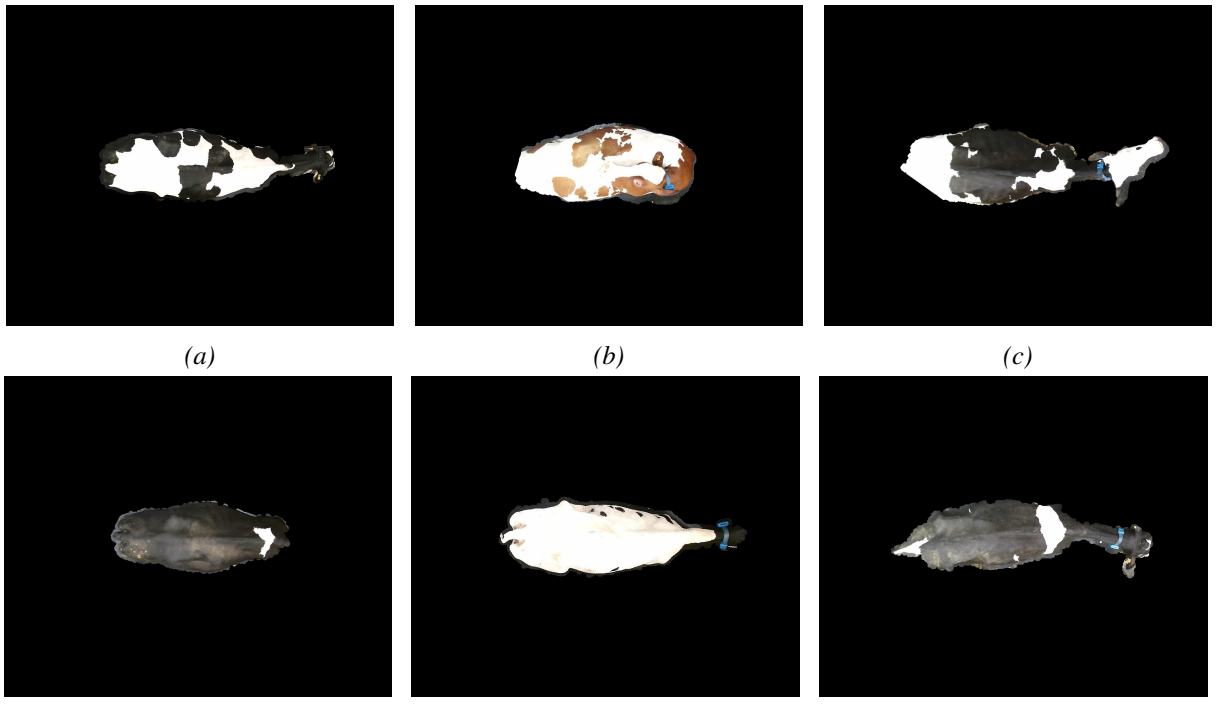
Figure 3.2: ***Raw Capture Examples.*** *Raw RGB image examples captured by a Microsoft Kinect 2 camera affixed statically above the one-way walkway connecting milking files and holding pens at the University of Bristol's Wyndhurst Farm for veterinarian teaching and research in Langford Village, UK.*

determined minimum and maximum sensor distances $t_1 = 2m$, $t_2 = 3.4m$ respectively, then binarised such that silhouettes are generated for the cows in the frame. Second, erosion and dilation were used to perform hole-filling on these silhouettes.

The resulting intermediates contain clutter in the scene at the same height as the cattle as well as secondary cattle that are only partially in the camera's field of view. Connected component analysis was used to remove any blobs smaller than "cow size" from the camera's fixed viewpoint. Finally, PCA was applied to each blob individually and the major axis (first principal component) is utilised to rotate each of the masks and its corresponding RGB data such that it is aligned with the horizontal axis. The result being that images are normalised, containing a single horizontally-aligned cow segmented against the background with its head on the right-hand side. Fulfilment of this process yielded the dataset used here. The FriesianCattle2015 Dataset consists of 274 preprocessed images of 35 individuals and is published online[2]. Figure 3.4 provides example images from this dataset, whilst Figure 3.5 illustrates examples of preprocessing failure cases that are not included in the final published set.

Figure 3.3: ***Dataset Acquisition Location.*** *Aerial imagery of the University of Bristol's Wyndhurst farm in Langford Village, UK. Enclosed within the red perimeter is the extent of the farm's land and fields. The barn location, where indoor data was acquired for this chapter, is highlighted by the yellow marker. Also shown (left) is the location of the farm within the UK. Images courtesy of: Google Earth Pro.*

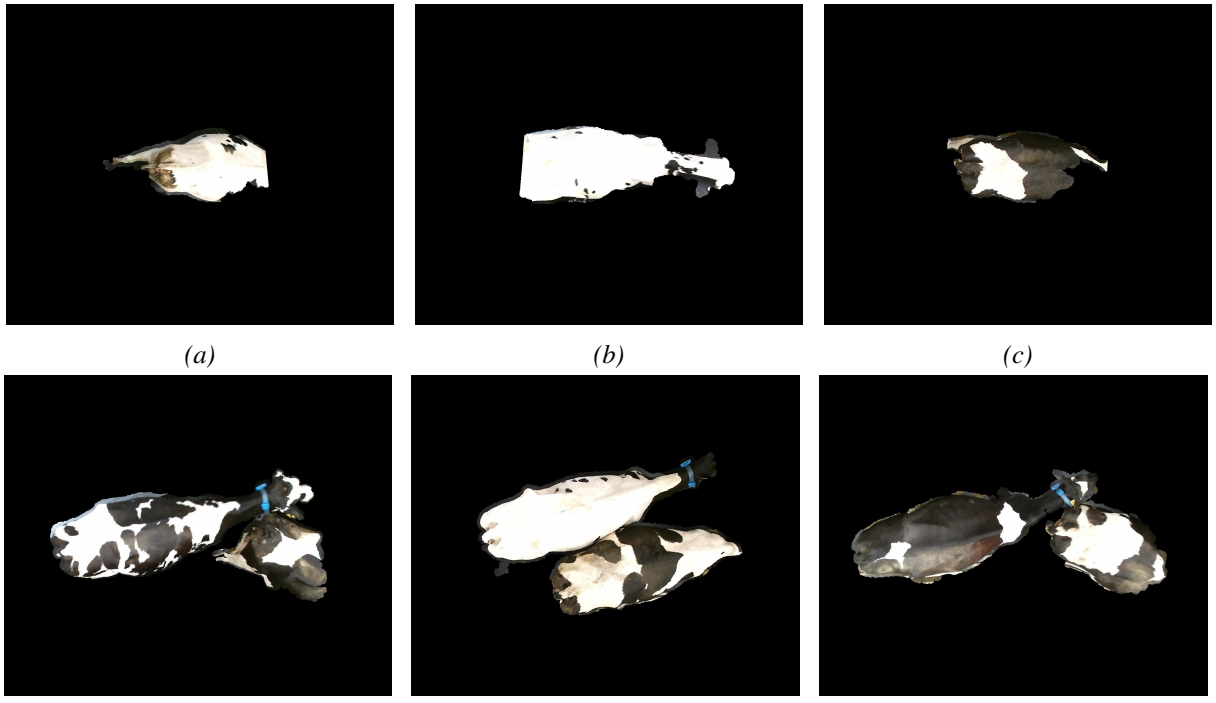
[2]FriesianCattle2015 dataset: https://data.bris.ac.uk/data/dataset/wurzq71kfm561ljahbwjhx9n3



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 3.4: ***Good Quality Segmentation Examples.*** *Example instances from the FriesianCattle2015 dataset that have been well preprocessed; constituting a single, horizontally-aligned cow segmented against the background. The occlusion of the head (visible in some cases) was deemed to be acceptable since they were observed largely not to contain visual features important for identification (as is indicated later in Figure 3.7) and introduce the potential for significant spatial variation.*



(a) *(b)* *(c)*

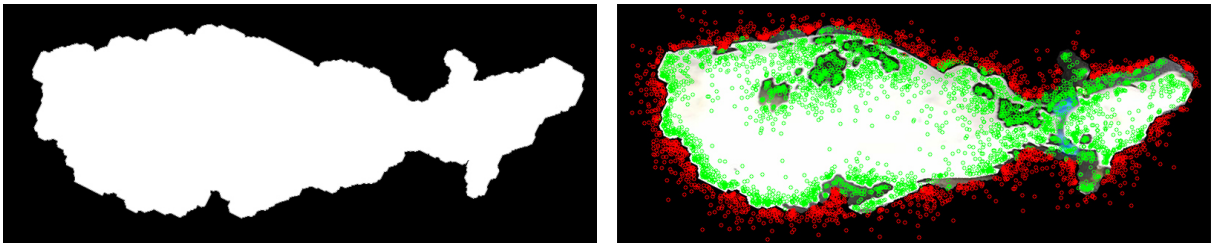
(d) *(e)* *(f)*

Figure 3.5: ***Erroneous Preprocessing Cases.*** *Examples of erroneous preprocessing cases – highlighting limitations of the approach – that were subsequently not included in the FriesianCattle2015 dataset due to the following reasons; First, (a)-(b): partial occlusion of the subject cow as a result of the individual walking out of the camera's field of view. Second, (c): poor depth segmentation resulting in significant partial occlusion. Finally, (d)-(f): multiple individuals present in segmentation due to their proximity during RGB-D acquisition.*

3.2.2 Implementation

Feature Extraction and Filtering

Feature extraction upon images is performed by the C++ distribution of ASIFT [223, 340], which, extends the vanilla SIFT algorithm [199] to account for all affine deformation components via simulating camera longitude and latitude coordinate manipulation to solve the two outstanding affine transformation components originating from camera tilt. The remaining 4 affine parameters are solved for by the SIFT algorithm itself. Whilst feature extraction could have been performed here instead via SIFT, it was deemed that effective tilt variation created by object deformations due to the animals walking, neck movement, etc. would be difficult to recover. Note however, that the choice in the utilisation of ASIFT instead of SIFT comes at the cost of significant additional computation time. Extracted ASIFT features are then filtered to limit features to have position within the animal area by discarding features spatially situated outside the segmentation boundary as exemplified in Figure 3.6. This stage is performed in order to make steps towards removing erroneous visual features created artificially by the segmentation boundary itself. This is achieved by overlaying ASIFT feature $f \in Cow_i$ coordinates f_x, f_y in the composited image (see figure 3.4 for examples) onto the corresponding mask image MI (see figure 3.9a) yielded by the aforementioned preprocessing stage (see section 3.2.1). If the pixel value at that coordinate is white $MI[f_x, f_y] = 255$, feature f is retained, otherwise it is discarded.



(a) Object mask image MI *(b) Composited RGB object image*

Figure 3.6: ***Extracted and Filtered ASIFT Features.*** *Mask image (a) pixel values $MI[x,y]$ are used to retain (green) features spatially positioned within the segmentation boundary for the composited image (b). Features located outside the animal region are discarded (red).*

Species-Specific Model of Descriptor Individuality

Many of the extracted and filtered ASIFT features in the animal region still carry little or no information about the identity of the individual itself. Put differently, features may be spatially situated in plain/monotonal areas carrying no individual structures, encode the highly variable silhouette, the tail or other non-individual features, markings or visual structures. To learn which subset of feature descriptor structures is individually characteristic or, put differently; contributes beneficially towards individual identification, a RBF-SVM was trained. Specifically, there was a desire to predict features as either individually characteristic or not based on the structure of the associated 128-wide descriptor vectors alone. The associated binary ground truth was obtained by placing a threshold on the term $D_f \in [-1, 1]$ for a feature $f \in Cow_i$:

$$D_f = \frac{|M_{intra}|}{|I_{intra}|} - \frac{|M_{inter}|}{|I_{inter}|}, \tag{3.1}$$

where M denotes pairwise feature matches for feature f for matching an image pair of the same individual (intra) or different individuals (inter). Lastly, I denotes the set of all training pairs. Accordingly, high D_f values denote that feature f is distinctive to its class and vice versa. A suitable threshold t was determined

following D_f computation upon the training data set. ASIFT features with $D_f > t$ and $D_f \leq t$ were binarily labelled positively and negatively, respectively. The feature training set of $\sim$7,000 image pairs (of individuals not used in later testing) together with this supervisory data was subsequently provided for SVM training. A grid search was performed a priori in order to determine suitable SVM training values resulting in $C = 2$ and $\gamma = 0.25$. 10-fold cross validation was subsequently completed, achieving 85.6% accuracy on the training data. Highest-performing threshold value $t = 0.18$ on D_f was then selected qualitatively as a result.

Resulting feature-wise classifications are depicted in figure 3.7, where the effectiveness of this model is clearly visible in removing segmentation boundary features whilst retaining those situated around individually-unique dorsal markings. The trained SVM is utilised by the subsequent feature matching stage to predict the importance and direct the inclusion of features for consideration during identity recovery.

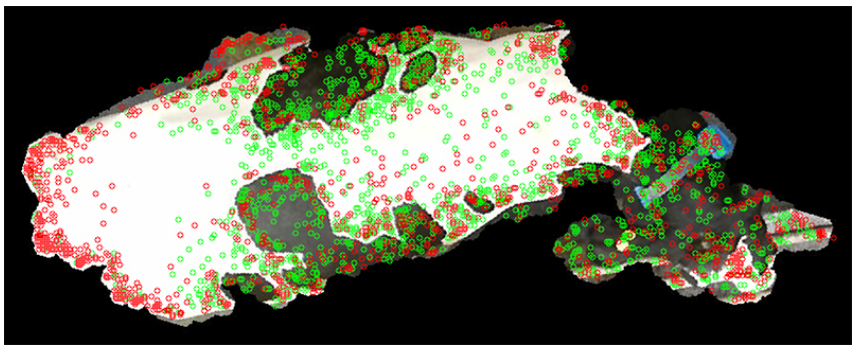


Figure 3.7: ***Application of Individuality Model.*** *Feature acceptance (green) and rejection (red) following thresholding on respective feature D_f values with $t = 0.18$. Rejected features can be seen to typically sit near highly variable segmentation boundaries.*

Feature Matching and Identification

Image-image comparison employing ASIFT feature matching is performed by its released C++ implementation[3]. Following feature extraction and filtering via the processes described in previous Section 3.2.2, retained features are matched for some image-image pair via the algorithm described in the original ASIFT article [340]. This process involves utilisation of the SIFT feature matching method [199] followed by filtering out matchings that are not consistent with an epipolar geometry via the Optimized Random Sampling Algorithm (ORSA) method [221].

Matching results are sequentially produced by performing feature-feature matching upon all possible image pairs. The results are comprised of the number of common features (matches) found for a particular image pair following aforementioned filtering. That is, *(a)*: enforcing features to reside within the animal region, *(b)*: filtering non-contributing features via the individuality model and *(c)*: verifying matches geometrically described as follows.

Image-image matches are verified geometrically by aligning image pairs vertically (see figure 3.8, row 4; lines between corresponding features forming a match are rendered in red). Only matches/lines which are less than $\pm 3°$ off the median are retained. This equates to a basic, relaxed linear model, which, was found to significantly improve true positive and negative identification success rates. Subsequent to filtering using the trained SVM, features comprising a match for an image pair are binarily classified $\in \{-1, 1\}$. Matches where both features are predicted to be characteristic to cattle via the individuality model are retained.

[3] ASIFT source code and articles: `http://www.ipol.im/pub/art/2011/my-asift/`

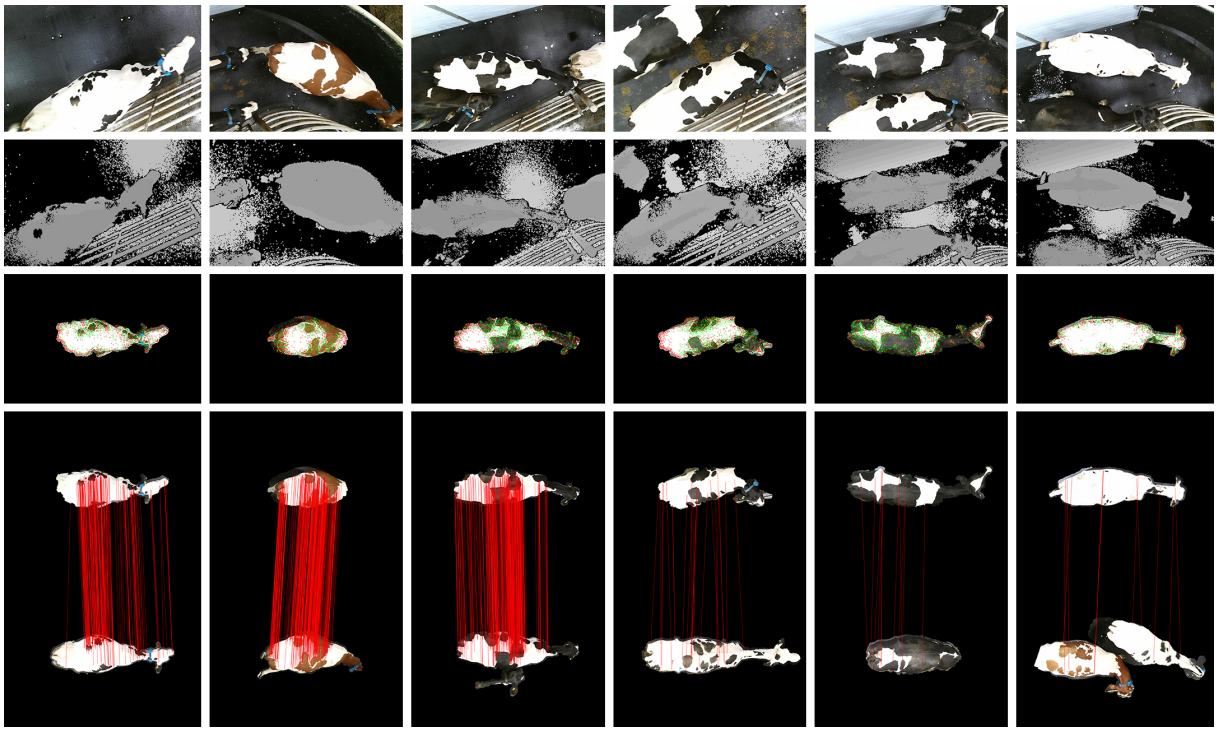


Figure 3.8: ***Examples of Identification Process with Successful and Unsuccessful Image-Pair Comparisons.*** *Row 1: RGB images and row 2: corresponding depths image. Row 3: images yielded from preprocessing with retained and discarded features following feature-importance prediction in green and red respectively. Rows 4, 5: examples of feature matching and geometric filtering on the same individual (left 3 examples) and different individuals (right 3 examples).*

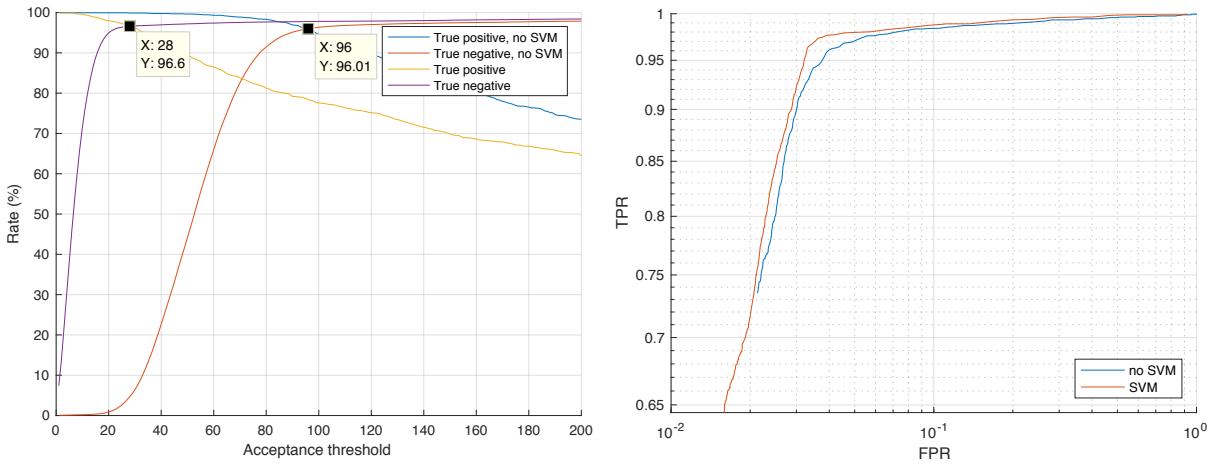
3.2.3 Experiments

Performance analysis for individual identification is accomplished by applying a threshold to the matchings quantity matrix (each test image vs. each test image without self-comparisons). Entries into this matrix contain the number of matched features for every possible image pair following aforementioned filtering. This threshold is varied to observe the effect upon true positive and true negative identification success rates as illustrated in Figure 3.9.

To showcase the improvements to scalability via the full individuality model in the proposed approach, two datasets are generated for training and testing arbitrarily from the larger FresianCattle2015 dataset. A subset of the collection is used in order to reduce computational expense. Figure 3.9 illustrates comparative results via Receiver Operating Characteristic (ROC) curves. The training dataset sampled from the FresianCattle2015 dataset consists of 10 individuals, 83 images and the testing dataset consists of 25 individuals, 191 images – yielding $191^2 = 36,481$ testing image-image pairwise comparisons. At Equal Error Rate (EER), the model achieves 96.6% identification accuracy on this data (accompanied by 69% feature-importance prediction accuracy). Disabling the individuality model was observed to result in a small decrease in identification accuracy as given by the ROC curves of Figure 3.9b. Figure 3.9a illustrates a significant reduction in the threshold on the quantity of mutual features for an image-image pair to be considered a match, whilst exhibiting a slight increase in overall accuracy (96.6% versus 96%). This highlights the ability of the individuality model in identifying and discarding features that do not beneficially describe dorsal coat patterns.

One important consideration is the computational cost of this implementation. Feature extraction alone (generating ASIFT feature descriptors) was found to require approximately 30 seconds *per image*. The addition of subsequent identification implementation processes further deteriorates overall identity recovery time – rendering it computationally infeasible in online scenarios (e.g. online, near real-time

identification operating on-board a UAV). Whilst efforts have been made in the parallelisation of ASIFT [58, 310], performance remains insufficient when coupled with the processes required with the identification pipelines described here in this thesis. Consequently, the testing population size and number of images per individual are arguably limited in these experiments. Whilst this serves well as a proof-of-concept experiment, it ultimately highlights the unsuitability of this approach in a live setting required in this work.



(a) *(b)*

Figure 3.9: ***Experimental Results.*** *(a): True positive and true negative rate vs. the feature acceptance threshold (minimum number of acceptable pairwise feature matches) with and without using the individuality model (marked no SVM). (b): ROC curves marginally confirm the effectiveness of the individuality model (orange) vs. basic ASIFT matching (blue).*

3.2.4 Section Conclusion

This section finds that for an arbitrary subset of the larger FriesianCattle2015 dataset – imagery as can be routinely generated in farm environments – it can be concluded that the application of an individuality model for filtering local descriptors is beneficial. The employed pipeline led to a best accuracy of 96.6% over the small testing set. More widely speaking, this clearly demonstrates that Holstein Friesian dorsal coat patterns are sufficiently visually distinctive/individual for identification purposes given a small population. This success may be subject to a caveat involving sufficient individual resolution and general image quality. This section has demonstrated that the proposed approach scales well across small herd sizes but, as mentioned previously, suffers as a result of the high computational cost associated with ASIFT feature extraction per-image. This renders identity evaluation of large population sizes computationally infeasible and justifies the implementation of a contemporary deep learning approach described in the following section. Correspondingly, future work could quantify the tradeoff between identification accuracy and execution speed via employing alternative feature descriptors (e.g. SIFT [199], SURF [26], ORB [268]). Furthermore, the method proposed here performs verification of whether two presented images contain the same individual or not. In operating individual identification on-board a robot agent, verifying whether a current image matches some template knowledge is inefficient, despite possible search acceleration via k-d trees [28] or similar. The task at hand is intrinsically a classification task; the UAV agent acquires an image and seeks to infer the identities of cattle present within it.

3.3 Identification via Deep Learning

In this section, a demonstration that computer vision pipelines utilising deep neural architectures are well-suited to perform automated individual identification of Holstein Friesian cattle in an agriculturally relevant setup is given. This is achieved via demonstrating that off-the-shelf networks can perform robust end-to-end identification of individuals in top-down still imagery acquired from fixed cameras. Additionally, this section introduces and operates upon a new dataset named FriesianCattle2017 of in-barn top-down imagery. This section shows that an individual identification model fuelled by a R-CNN exploiting dorsal Friesian coat uniqueness on 940 RGB stills (containing 89 unique individuals) taken after milking in-barn achieves an accuracy of 86.1%. This test suggests that, an application of marker-less Friesian cattle identification is not only feasible using standard deep learning components – it appears robust enough to assist existing tagging methods. At the time of writing, this work was the first to apply deep learning to the task of automated visual bovine identification.

3.3.1 Individual Identification Implementation

In order to achieve the goal here of individual Holstein Fresian identification via deep learning, a R-CNN ANN/DNN architecture is employed. Specifically, the employed network architecture is a R-CNN adaptation of the VGG-M 1024 CNN described alongside a series of other proposed architectures [47]. Acquired images are passed through the R-CNN ultimately resulting in a set of estimated object bounding boxes and respective identity estimates. Whilst a simpler CNN could have been employed solely on provided object regions, the benefit of individual detection and localisation occurring as in implicit process is beneficial in applicational settings due to computation time and otherwise. What perhaps this merging of processes fails to account for is circumstances involving a new, previously unseen individual. In this case, the target is perhaps unlikely to be detected whereas a robust species-wide detector would likely provide the desired result. The simple image to ID pipeline is illustrated in Figure 3.10.

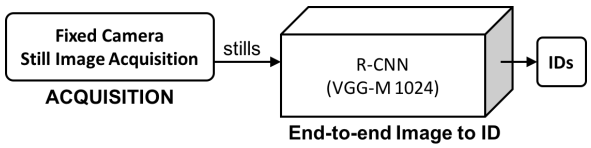


Figure 3.10: ***Identification and Implicit Detection Pipeline.*** *Off-the-shelf baseline layout utilising still image input and comprising a VGG-M 1024 [47] R-CNN trained end-to-end for individual Holstein Friesian cattle identification and localisation.*

Formalisation of this identification and implicit detection and localisation task begins with defining the possible classes for proposed object RoIs yielded from the R-CNNs RPN:

$$C_{ID} = \{background, cow_0, cow_1, ..., cow_m\}, \tag{3.2}$$

where $|C_{ID}| = m+1$ to include the negative class case. Inference on an image I with the R-CNN yields a set of n bounding box rectangles:

$$B = \{bbox^1_{pred}, bbox^2_{pred}, ..., bbox^n_{pred}\} \tag{3.3}$$

defined spatially by the tuple:

$$\begin{aligned} bbox^i_{pred} &= ((x^i_1, y^i_1), (x^i_2, y^i_2)), \\ &\text{where } \forall\, 0 < i \leq n. \end{aligned} \tag{3.4}$$

Also, associated with each predicted object RoI is a class membership probability vector yielded by the network:

$$P = (p_0, p_1, ..., p_n),$$
$$\text{such that } \sum_{i=0}^{n} p_i \in P = 1,\ |P| = |C_{ID}| \ \text{ and } \ \forall p_i \in P,\ p_i \in \mathbb{R}^+, \tag{3.5}$$

as enforced by a softmax function. The final content prediction c^i_{pred} of a predicted RoI $bbox^i_{pred}$ is taken to be the maximal element of P:

$$c^i_{pred} = \underset{p \in P}{\text{argmax}}, \tag{3.6}$$

where $c^i_{pred} \in C_{ID}$. Note that class *background* is somewhat ambiguous; it potentially now incorporates a new, unseen individual not typically associated with the background itself. The hope is that given sufficiently strong training, such a case is not paired with an erroneous identity $c \in C_{ID} - background$ or is not detected whatsoever.

3.3.2 Dataset: FriesianCattle2017

This dataset consists of 940 RGB images based on data capture carried out in the previous Section 3.2.1. Of those images, there are $|m| = 89$ distinct Holstein Friesian individuals such that:

$$C_{ID} = \{background, cow_0, cow_1, ..., cow_{88}\}$$
$$\text{and } |C_{ID}| = 90. \tag{3.7}$$

The data was captured over a two hour-long session via the use of a Microsoft Kinect 2 camera affixed statically over the walkway between holding pens and milking stations in a real indoor farming environment. The camera was configured to capture top-down still images of cattle dorsal coat patterns at a rate of 0.5 Hz. Whilst data capture could have instead been triggered via depth sensing of appropriately sized and placed blobs – potentially removing the occurrence of individuals partially outside of the image frame – depth sensing capabilities are not equipped on-board the UAV agent performing individual identification in Chapter 7 and it is desirable to replicate this scenario as closely as possible in preparation. A more in-depth discussion of dataset acquisition, preprocessing steps, etc. is given in the aforementioned section 3.2.1. Example images of individuals from the dataset are given in Figure 3.11.

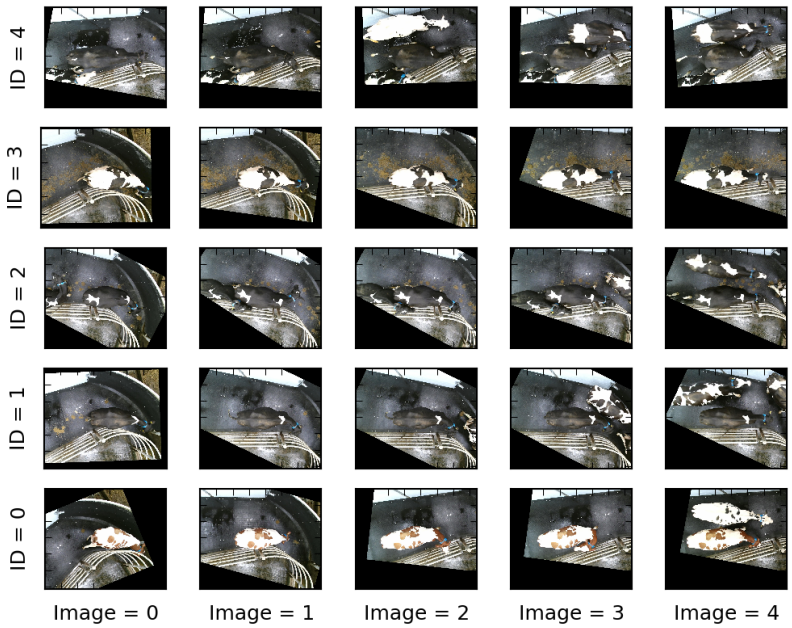


Figure 3.11: ***Indoor Dataset Example Instances.*** *Example rotation-normalised still images from the indoor dataset used in this and the previous section. (rows): Example individuals and (columns): example instances (or frames) for a particular individual from the FriesianCattle2017 dataset. The camera was affixed statically above a walkway between holding pens and milking stations in a real agricultural environment.*

Ground-Truth Labelling

The use of a R-CNN intrinsically requires training data to include ground truth object bounding boxes as well as class labels for each object RoI. The dataset ground truth labelling process was therefore a two-fold task:

1. ***Bounding Box Annotation***: After user annotation of bounding boxes in an image, the generated data was stored in an XML format aligned with that required by the Faster-R-CNN framework and Convolutional Architecture for Fast Feature Embedding (Caffe) [150] for instance annotation. Annotations were performed manually in adherence with the labelling guidelines of the VOC challenges [92] – specifically, the VOC2012 guidelines [93]. Manual annotations were accomplished sequentially utilising a custom-built Graphical User Interface (GUI) application (see Figure 3.12) where users can "click and drag" rectangles over object RoIs.
2. ***Individual/Class Labelling***: Following bounding box annotation, the users were sequentially presented each labelled RoI and asked to identify the individual cow contained within the presented bounding box (see Figure 3.13).

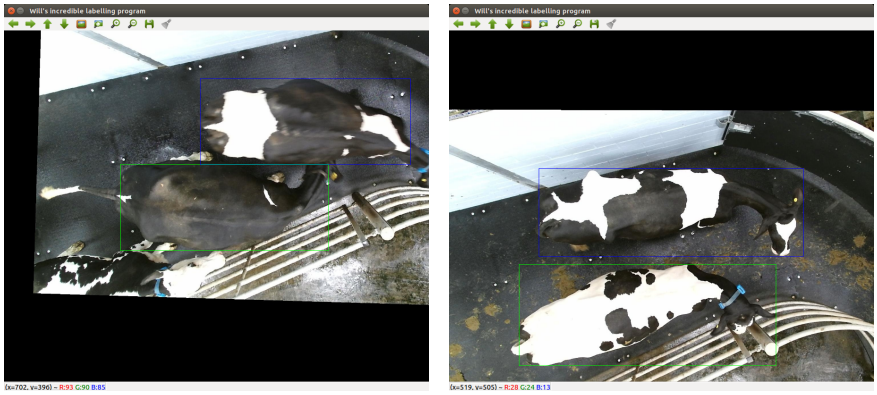


Figure 3.12: ***Bounding Box Annotation Program.*** *Custom-built GUI application for ground-truth object bounding box annotation written in OpenCV [147]. For each dataset image, the user can manually "click and drag" a rectangle over individual cattle RoIs (green rectangles) in accordance with the VOC2012 labelling guidelines [93] and accept (blue rectangles) or re-draw their input.*

(a) Labelling query example with differing individuals and therefore, the user would press key 'n'.

Figure 3.13: ***Class Labelling Program.*** *Screenshot from the GUI application designed to aide the process of categorising image instances into individuals so as to generate ground-truth individual identity data. The user is presented with two green bounding boxes side-by-side yielded from the labelling process depicted in Figure 3.12 and is asked whether the two bounding boxes contain the same individual or not.*

Labelling Challenges

It was found that in a significant proportion of dataset images, one or more individuals would be partially occluded; either by another individual or being spatially positioned partially outside of the image frame boundaries. This creates a fundamental and philosophical question with respect to the manual ground truth labelling stage; how visible should an object be for the user to label it? More abstractly, what constitutes valid parameters for object visibility? Specific to the use case here, in some instances, an individual's tail was found to be somewhat upwards and away from the body. This poses the question: should the ground truth bounding box annotation include the tail in these situations? Its inclusion would imply that many erroneous background pixels are also included within the annotation. The same problem applies equally for a cow's head and legs/feet to a certain extent.

Fortunately, these questions and problems are answered by guidelines created for the VOC challenges [92]. For ensuring consistency in ground truth dataset labelling, all instances are labelled to adhere to the guidelines listed below for the entirety of this thesis. In particular, the VOC2012 guidelines [93] state:

- ***What to label***: all objects of the defined categories, unless: you are unsure what the object is, the object is very small (at your discretion) or less than 10-20% of the object is visible.
- ***Bounding box***: mark the bounding box of the visible area of the object (not the estimated total extent of the object). Bounding box should contain all visible pixels, except where the bounding box would have to be made excessively large to include a few additional pixels (<5%) e.g. a car aerial.
- ***Occlusion/truncation***: if more than 15-20% of the object is occluded and lies outside the bounding box, mark as "Truncated". Do not mark as truncated if the occluded area lies within the bounding box.

Data Augmentation

During data capture, cows were free to walk from a holding pen to a milking station as they pleased (see section 3.2.1 for more details on data acquisition). This results in per-individual variation in the time spent within view of the static acquisition system. Consequently, the number of images (or instances) obtained per individual are not balanced across the population – the mean being $\mu = 15$ images with standard deviation $\sigma = 19.9$. Figure 3.14 shows the distribution of instances per individual over the data acquisition period.

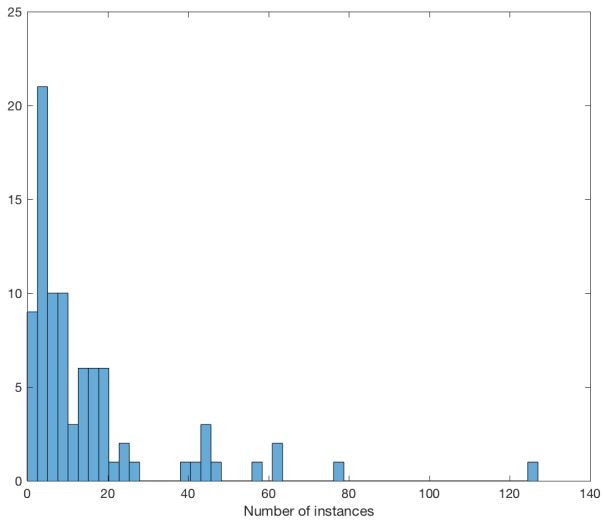


Figure 3.14: ***Instance-Individual Distribution.*** *Histogram of the number of original (non-synthesised) instances (images) per individual cow ($\mu = 15$, $\sigma = 19.9$).*

To balance the number of images per individual for training purposes, the dataset was augmented via image synthesis. The target number of instances was chosen to equal the maximum number of original (non-synthesised) instances for any particular individual (in this case, 127 images for Cow 11). Additional images were synthesised by rotating original images by some random angle α about the image centre (x_c, y_c) whilst maintaining the original image resolution for dataset consistency. Bounding box coordinates/parameters were also transformed by angle α. New bounding box coordinates (x_b, y_b) were computed via:

$$\begin{aligned} x_b &= (x_a - x_c)cos\alpha - (y_a - y_c)sin\alpha + x_c \\ y_b &= (x_a - x_c)sin\alpha + (y_a - y_c)cos\alpha + y_c \end{aligned} \tag{3.8}$$

where (x_a, y_a) denotes bounding box coordinates prior to image rotation and (x_c, y_c) denotes the image centre. This stage yields an angled bounding box annotation for the respective object outline. However, since the Faster R-CNN implementation currently does not support the parametrisation of object rotation (bounding box angle), an orthogonal bounding box is generated via min, max functions of transformed coordinates. That is, for a bounding box defined spatially by $bbox_a = ((x_a^1, y_a^1), (x_a^2, y_a^2))$ which is rotated by random angle α to become $bbox_b = ((x_b^1, y_b^1), (x_b^2, y_b^2))$, the resulting image-frame-orthogonal bounding box $bbox_c$ is generated by the following equations for a top-left image origin:

$$\begin{aligned} x_c^1 &= min(x_a^1, x_b^1) \\ y_c^1 &= min(y_a^1, y_b^1) \\ x_c^2 &= max(x_a^2, x_b^2) \\ y_c^2 &= max(y_a^2, y_b^2) \end{aligned} \tag{3.9}$$

The negative implication of this is that more erroneous background pixels are often included within ground truth object RoIs as a result of cattle being largely rectangular in shape. Furthermore, transformed bounding box coordinates were bounded within the dimensions of the image frame.

Despite the synthesis process, the resulting augmented dataset was not altogether perfectly balanced (equal numbers of instances for all individuals). This is since many of the original images contain more than one individual. When synthesising new images with multiple cows present originally, possible surplus instances for other individuals are created. Whilst this problem could easily be avoided by only synthesising on images with one individual present, a significant proportion of original instances contain multiple individuals. To only synthesise on such images would effectively reduce the scope of the training data and potentially lead to model over-fitting.

To re-balance the augmented dataset following synthesis, a simple algorithm is employed whereby surplus instances are removed. If a particular individual has a surplus, synthesised instances are sequentially inspected and potentially deleted until a target class cardinality is reached. For an individual with a surplus, synthesised instances are deleted if they contain only one individual. In the case that the instance being inspected contains multiple individuals whom also all have a surplus, the instance is marked for deletion. This is not a perfect solution, since the distribution of individuals within instances is non-uniform. The yielded dataset following this process is therefore likely to still be unbalanced, but nevertheless an improvement.

Training-Testing Data Partitioning

Following synthesis, the yielded dataset is partitioned into training/testing sets towards the goal of performing two-fold cross validation. Algorithm 1 is employed for partitioning the dataset into appropriate bins since it would be inappropriate to evaluate the network's performance on synthetic instances; images that have already been trained upon in their original form.

Algorithm 1 Training/Testing Data Partitioning. Algorithm for splitting datasets comprised of synthetic and original instances into randomly generated bins for training and testing towards k-fold cross validation. Note that the functions: $getInstances(x)$ and $getSyntheticInstances(x)$ return the list of original (non-synthesised) and synthetic instances respectively for individual x.

$folds \leftarrow 2$
$max \leftarrow findMaxInstances(individuals)$
for ind **in** $individuals$ **do**
    $bin\_size \leftarrow \frac{|getAllInstances(ind)|}{folds}$
    $non\_synth \leftarrow randomShuffle(getInstances(ind))$
    $synth \leftarrow randomShuffle(getSyntheticInstances(ind))$
    if $|non\_synth| < bin\_size$ **then**
        $non\_synth \leftarrow splitIntoBins(non\_synth, folds)$
        for i **in** $folds$ **do**
            $label\_data[ind][i].test \leftarrow non\_synth[i]$
            $label\_data[ind][i].train \leftarrow synth + non\_synth[0:i-1] + non\_synth[i+1:end]$
        end for
    else
        $all\_instances \leftarrow non\_synth + synth$
        $all\_instances \leftarrow splitIntoBins(all\_instances, folds)$
        for i **in** $folds$ **do**
            $label\_data[ind][i].test \leftarrow all\_instances[i]$
            $label\_data[ind][i].train \leftarrow all\_instances[0:i-1] + all\_instances[i+1:end]$
        end for
    end if
end for

3.3.3 Single-Frame Individual Identification

For this task, the augmented (including synthesised images) FriesianCattle2017 dataset of indoor still images was used. Following synthesis on the 940 original instances for 89 individuals, the ∼11,000 yielded instances were randomly segmented towards two-fold cross validation as per algorithm listing 1. However importantly, synthesised instances were never included in testing sets (see Section 3.3.2). As mentioned previously, the VGG M 1024 CNN [47] network adapted for R-CNN is employed here. Training end-to-end for 100,000 iterations and a batch size of 32 for both folds, the mAP values given in Table 3.1 were obtained. Figure 3.15 illustrates occasions where identification failed, producing incorrect results. These were found to be due to observable visual similarity across individuals, multiple cow proximity/alignments and frame boundary clipping.

Note that region predictions from the R-CNN are accepted as true positive provided there is sufficient overlap with a same-class ground truth bounding box via a binary threshold $t = 0.5$ placed on scalar value ov describing rectangle-rectangle overlap via Intersection over Union (IoU):

$$ov = \frac{bbox_{GT} \cap bbox_{pred}}{bbox_{GT} \cup bbox_{pred}} \tag{3.10}$$

where $bbox_{GT}$ and $bbox_{pred}$ denote the ground truth and predicted bounding box regions respectively. Categorised detections are subsequently used to compute class precision and recall data and mAP is computed via the corresponding Area under Curve (AuC).

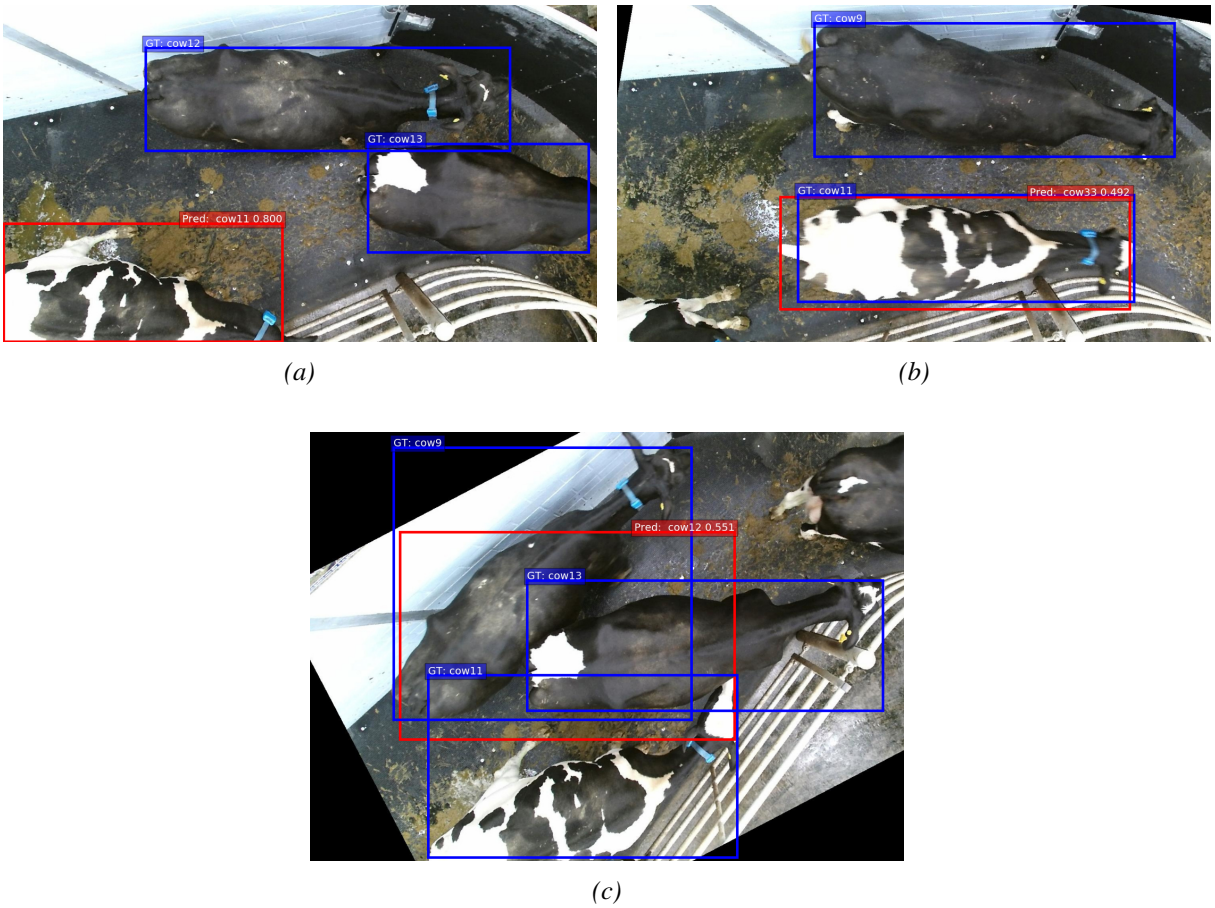


Figure 3.15: ***Single-Frame Identification Failures.*** *Shown are ground-truth/prediction examples where single-frame cattle identification failed due to: (a) no corresponding ground-truth annotation after following VOC labelling guidelines [92, 90], (b) individual visual similarity and (c) region proposal error due to multi-cow alignment/proximity.*

Task	mAP (%)		
	Fold 1	Fold 2	Average
R-CNN Identification & Localisation	87.21	84.93	**86.07**

Table 3.1: ***Single Frame Individual Identification Accuracy.*** *Classification mAP for single frame individual identification and localisation on the indoor FriesianCattle2017 dataset over 2-fold cross validation via the use of the VGG M 1024 CNN [47] architecture.*

3.4 Comparative Study

The two approaches employed towards identifying individual cattle from single images are fundamentally at odds operationally. The result is that directly comparing the performance of the two approaches is not necessarily fair. On the one hand, the classical approach applies feature-feature matching between template and query image pairs, whereas convolutional networks infer identity from single images and require a distinct supervised learning phase. This section, however, provides a limited comparison by providing results on one-shot learning using a CNN and image augmentation on the very same dataset used for the former, classical approach.

Imagery used here is the testing set from the FrieisanCattle2015 dataset (refer back to Section 3.2.1 for

more details) consisting of 191 preprocessed images of 25 individuals. These are the exact same images that the local feature matching approach used for testing. A single image was randomly subtracted from each of the 25 categories to seed one-shot learning, meaning that yielded results are not directly comparable. These images were then augmented[4] with a combination of random cropping, scaling, rotation, blurring and more to produce sufficient levels of training data, with 500 instances per category (including the original non-augmented image). The resulting training dataset was then used to train the VGG CNN-M architecture over 50 epochs to provide a fair comparison, identically to the R-CNN architecture used in earlier Section 3.3 minus Region Proposal Network (RPN) components. The network architecture is fully illustrated in Table 3.2, whilst Figure 3.16 illustrates accuracy versus training steps, where 10% of the training set was retained for validation.

CNN-M	96x7x7 st. 2, pad 0 LRN, x2 pool	256x5x5 st. 2, pad 1 LRN, x2 pool	512x3x3 st. 1, pad 1 -	512x3x3 st. 1, pad 1 -	512x3x3 st. 1, pad 1 x2 pool	4096 drop- out	4096 drop- out	1000 soft- max

*Table 3.2: **VGG CNN-M Architecture.** The VGG CNN-M architecture used here to perform a comparative study between the two approaches to individual cattle identification proposed in this chapter. The architecture is published as part of several other network proposals in the original paper [47]. The network architecture is identical to the one used in the earlier proposed deep learning approach (see Section 3.3) employing Faster R-CNN, except that it is used here as a standard CNN (i.e. the RPN is removed and the whole image is classified). Table credit: Chatfield, K. et al. [47]*

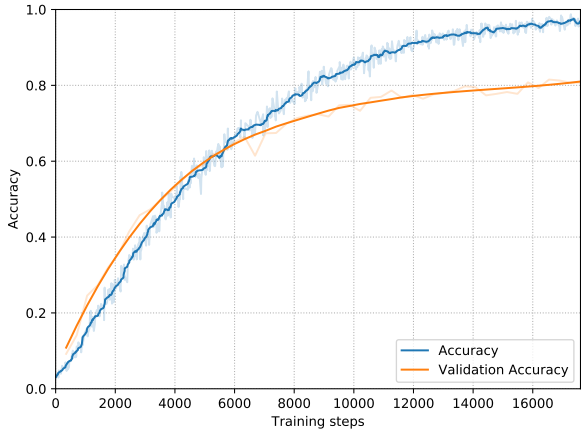


*Figure 3.16: **Accuracy vs. Training Steps.** Graph of training and validation set accuracy[5] versus training steps. The VGG CNN-M [47] architecture was trained for 50 epochs on 500 synthetic augmented images from a single image source for each category (including the one original non-augmented seed image). As can be seen, significant overfitting of the training set is observed to occur.*

Comparative results, indicated in Table 3.3, demonstrate that the feature-feature matching approach outperforms one-shot learning, despite operating on a marginally larger testing set. The problem lies in the fact that all augmentations are seeded from a single image for each category, where the individual will be in some particular articulation/pose that is replicated over 499 instances. The resulting trained CNN will not be invariant to change in individual pose, as is regularly observed in the testing set where individuals walk in and out of frame. This highlights the importance of sufficient quantities of variant labelled training data for representation learning. Where this is difficult to acquire, and identification times are not an issue, local feature matching appears to offer more accurate end results over small populations.

Approach	Test Instances	Accuracy (%)
Local Feature Matching	191	96.6
One-Shot Learning CNN	166	62.65

*Table 3.3: **Comparative Performance Results.** Quantification of comparative performance for the two approaches to individual identification proposed in this chapter. Note that the ANN-based approach utilises a subset of the dataset for model training, resulting in fewer testing instances, meaning that the results are not directly comparable.*

[4] Images were augmented using: https://github.com/aleju/imgaug

[5] Noisy signals have been smoothed using the Savitzky-Golay filter [275].

3.5 Chapter conclusion

This chapter demonstrates that the task of automatic individual Holstein Friesian identification exploiting dorsal coat pattern uniqueness can operate well under a single-frame, single iteration evaluation paradigm. Importantly, this process can take place with minimal intrusion in practically relevant settings, in contrast to the majority of existing identification frameworks revolving around physical animal tagging.

Specifically, this task is achieved via a traditional descriptor approach followed by a contemporary approach founded in deep neural architectures. With respect to the former – employing ASIFT feature description and matching on dorsal coat patterns – identification performance is strong across a small-sized herd. The computational cost however, is significant to the point where it is infeasibly applicable in online settings, as required by this thesis in performing *live identity estimation* on-board a UAV robot agent.

This problem is alleviated by the latter approach; training a R-CNN to not only identity individuals, but also to detect and locate them in real-world imagery in an agriculturally-relevant environment. Across a larger population (consisting of 89 individuals), accuracy is strong, whilst per-image inference time is in the manageable region of sub one second for online computation. As a result of identification success, convolutional-based architectures are shown to be well suited towards learning and distinguishing the properties of unique dorsal pattern and structure exhibited by the species individually. Whilst results obtained here are strong under the paradigm of evaluating a single image, the following chapter will go on to show that: identification estimates yielded from relevant models benefit from exposure to multiple frames containing the individual in question under observation parameter variation.

Supplementary Information

Publications associated with this chapter:

1. W. Andrew, S. Hannuna, N. Campbell, T. Burghardt. Automatic Individual Holstein Friesian Cattle Identification via Selective Local Coat Pattern Matching in RGB-D Imagery. In *IEEE International Conference on Image Processing (ICIP)*, pages 484-488, 2016. https://doi.org/10.1109/ICIP.2016.7532404.
2. W. Andrew, C. Greatwood, T. Burghardt. Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning. In *IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 2850-2859, 2017. https://doi.org/10.1109/ICCVW.2017.336.

Accompanying published datasets available at:

- **FriesianCattle2015**: https://data.bris.ac.uk/data/dataset/wurzq71kfm561ljahbwjhx9n3
- **FriesianCattle2017**: https://data.bris.ac.uk/data/dataset/2yizcfbkuv4352pzc32n54371r

Chapter 4

Passive Multi-Frame Identification

4.1 Chapter Overview

Passively-acquired videos of individual cattle filmed by an aerial robot will naturally contain viewpoint, object and illumination variation. Considered in their entirety therefore, sequences of images contain complementary and superior information as opposed to single images. Given that operation on fine-grained categories necessitates the observation of subtle discriminative features, multiple variant frames increase the likelihood of their visibility altogether. In this chapter, this intuition is investigated in beginning to introduce multiple iterations to individual identification. This occurs here in a *passive* and *offline* setting, where the agent has no control over observation parameters in order to showcase the benefit of simplicity.

Passive iterative identification is achieved by a proposed video processing pipeline (see Figure 4.1 below) consisting of standard components to efficiently process dynamic aerial herd footage captured via the use of a UAV. The proposal consists of a trained species-wide detector feeding tracked individual-wise RoIs into a temporally-integrative architecture inferring cattle identity.

This chapter is organised firstly into Section 4.2 describing the acquisition and pre-processing of the dataset via the use of a UAV. Next, Section 4.3 outlines a component of the complete video processing pipeline performing species detection. Section 4.4 then details the iterative identification component of the pipeline, finishing with conclusions for this chapter in Section 4.5.

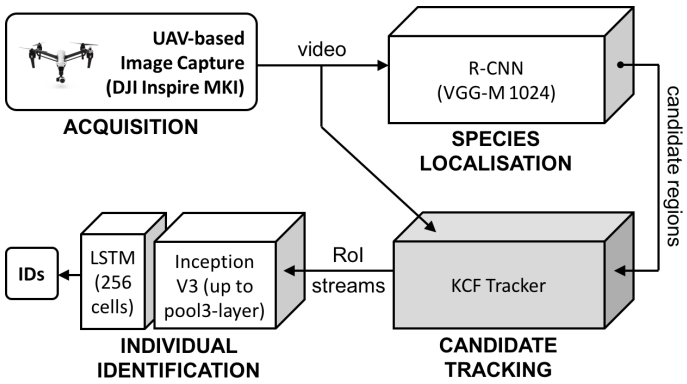


Figure 4.1: ***Proposed Iterative Identification Pipeline.*** *Aerial video processing pipeline with separation of species detection and individual identification components, added KCF [131] tracking unit for fast trajectory extraction, and enhanced temporally-integrating individual identification component combining an Inception V3 network [303] feeding into a standard LSTM unit [135] to utilise complementary information as revealed progressively across several frames in tracked RoI streams.*

*Figure 4.2: **UAV Acquisition of a Friesian Cattle Herd.** Representative frame from the AerialCattle2017 dataset captured by a DJI Inspire MkI UAV filming at ∼5 meters above the ground. Individual cows are resolved at approximately 300×100 pixels within frames resolved at 3840×2160 pixels captured at a frame-rate of 24 fps. The animals' distinctive black and white coat patterns are clearly resolved and are used as an individually-characteristic biometric entity.*

4.2 Dataset: AerialCattle2017

In order to train and evaluate the efficacy of the proposed approaches, this chapter utilises the AerialCattle2017 dataset captured in a practically relevant scenario[1]. This dataset is published online for public use[2]. The dataset consists of 34 herd videos of cattle captured from an aerial standpoint of grazing fields at the University of Bristol's Wyndhurst veterinarian farm in Langford Village, UK. Each video is approximately 20 seconds in length and is captured from a top-down perspective (see Fig. 4.2). For each video, cow regions were extracted and used to produce cropped videos containing single individuals (examples provided in Fig. 4.4). Following individual cropping, the resulting individual-centric RoI dataset contains 23 individuals and a total of 160 videos for a mean of $\mu = 7$ instances per individual with standard deviation $\sigma = 3.87$.

*Figure 4.3: **AerialCattle2017 Dataset Acquisition Location.** Aerial image[3] of the Wyndhurst Farm in Langford Village, UK. (Red): marked boundaries of the field where data acquisition, resulting in the AerialCattle2017 dataset, took place.*

[1] Many thanks to Professor Becky Whay for assisting with the capturing of data used in this chapter.

[2] AerialCattle2017 dataset: https://data.bris.ac.uk/data/dataset/3owflku95bxsx24643cybxu3qh

Figure 4.4: ***Aerial Dataset Examples.*** *Example frames from individual-centric RoI streams extracted from outdoor UAV-acquired video footage forwarded as input into the LRCN identity-recovery architecture. (rows) Example individuals (different RoI streams) and (columns) example instances (or frames) for that individual. The aspect to be noted is the variation exhibited across multiple frames for instances that reveal further salient identification features that the proposed architecture exploits (e.g. ID = 1). As well as this, the inclusion of erroneous pixels from other individual(s) can be overcome with information extracted from frames prior to the event (e.g. ID = 3).*

4.2.1 Acquisition

The aerial video dataset was acquired[4] via the use of a DJI Inspire MKI UAV, quadrotor or drone and its integrated camera/3-axis gimbal system (see figure 4.5), the DJI Zenmuse X3. It was flown above a herd of approximately 30 young Holstein Friesians in a nursery field (acquisition location depicted in Figure 4.3). During flights under the supervision and oversight of veterinarian researcher Professor

Figure 4.5: ***DJI Inspire MKI UAV[5].*** *Quadrotor platform manufactured by DJI utilised for UAV-based data capture. The Zenmuse X3 – the integrated 3-axis gimbal and 4K resolution camera – is situated at the bottom of the aircraft (for further camera specification details, see Section 7.2.1*

[3] Images courtesy of Google Earth Pro.
[4] Special thanks to Dr Colin Greatwood for piloting the UAV to acquire this dataset.
[5] Image credit DJI.

Becky Whay, footage was captured over an hour-long period at a raw resolution of 3840×2160 pixels at 24 fps. Over the 34 acquired videos, the UAV height from the ground was varied by 5 m decrements starting at 25 m and with the lowest height being 5 m – note that altitudes were varied only in-between video capture. This allowed the cattle to become gradually accustomed and comfortable with the physical and sonic presence of the UAV – after initially exhibiting signs of some anxiety towards it according to Professor Whay. Flights were performed in accordance with the Civil Aviation Authority (CAA)'s UAV regulations mandating maximal flight altitude, distance from properties/objects outside of the operator's control, etc. [16].

4.2.2 Ground Truth Labelling and Cropping

To produce ground-truth bounding box and class labels, the first frame of each video was extracted and considered as for the indoor, still-image dataset. Object bounding boxes were manually annotated for these frames in accordance with labelling guidelines created for the VOC challenges [90, 92]. As for the aforementioned FriesianCattle2017 dataset (see Section 3.3.2), labelled bounding boxes were subsequently manually categorised into respective individual classes using the labelling tool developed there (for illustrative examples, see Figure 3.13). To generate ground-truth bounding boxes for all subsequent frames of each video, an instance of the KCF tracking algorithm [131] was initialised for each RoI as per the pipeline depicted in Figure 4.1. Bounding box coordinates are updated over frames and were directly used to crop images to individual cow size. Clipped/lost regions were manually deleted to adhere to the VOC labelling guidelines. Cropped RoIs containing single cows were then saved as independent videos and grouped according to their original class label. Example frames yielded from this process are given in Figure 4.4.

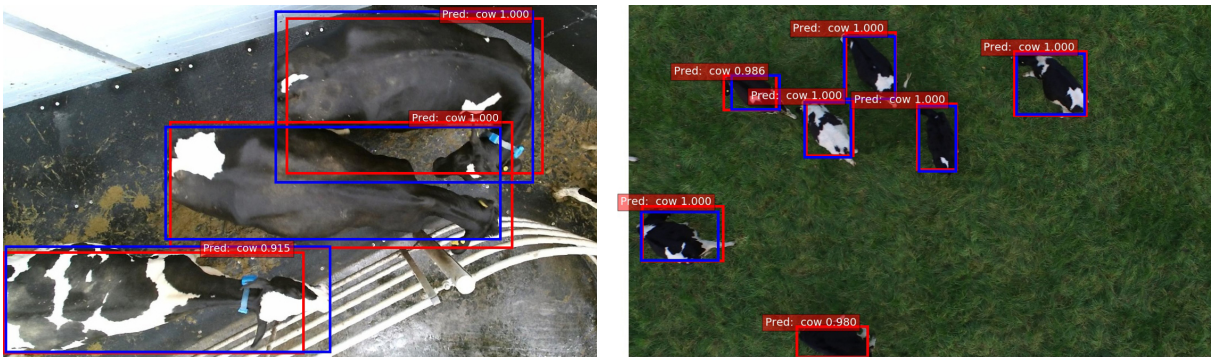
4.3 Species Detection and Localisation

The first component in the video processing pipeline architecture depicted in Figure 4.1 deals with detecting and locating Friesian cattle in given imagery. Candidate regions are then later provided as initialisation to individual-wise tracking components. In this section, an object detector is trained on:

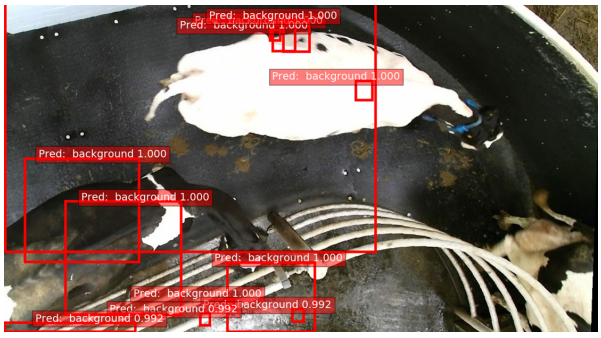
1. in-barn imagery from the previous chapter (the FriesianCattle2017 dataset, see Section 3.3.2)
2. frames extracted from outdoor, UAV-acquired video footage (the AerialCattle2017 dataset, see Section 4.2)

Figure 4.6 illustrates detection examples on both types of images. The intention is that in training on multiple image domains, where significant background, object, etc. variation occurs, the trained model generalises well to unseen images in learning domain-agnostic features.

The deep network utilised to address the goal of automated Friesian detection and localisation is the R-CNN adaptation of the VGG M 1024 CNN published as part of several other network architecture proposed by the Visual Geometry Group at the University of Oxford [47]. The core architecture consists of 5 stacked convolutional layers – which are shared with the RPN – plus three fully connected layers (the final with softmax activation). The architecture is illustrated in Figure 4.7. Instead of training from scratch, network weights were initialised with a model trained on the ImageNet database [66] supplied with the Faster-R-CNN Python implementation [259] founded within the Caffe framework [150] for ANN applications.



(a) Indoor detection examples. *(b) Outdoor detection examples.*



(c) Rejected background region proposals on an indoor dataset instance.

Figure 4.6: ***Species Detection and Localisation.*** *Examples of species detection and localisation for the (a): indoor FriesianCattle2017 and (b): outdoor AerialCattle2017 datasets using the R-CNN model as well as (c): examples of discarded background region proposals (i.e. ¬cow). (Blue): Ground truth bounding boxes for object class cow and (red): predicted candidate object bounding boxes with class membership score values.*

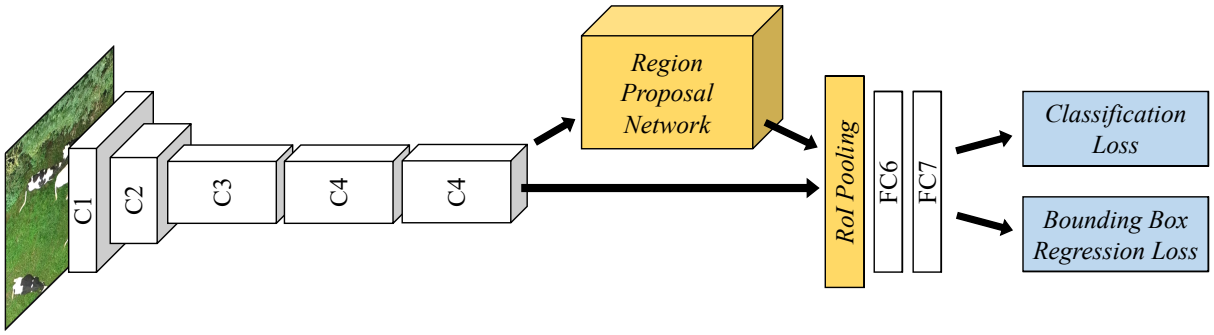


Figure 4.7: ***Species Detector Network Architecture.*** *The employed neural architecture for performing species-wide detection and localisation within two-dimensional imagery. The architecture is the VGG M 1024 CNN [47] adapted for faster R-CNN. Network output is a set of candidate object regions with associated classification scores for a given input image.*

4.3.1 Quantitative Findings

Cattle detection and localisation was implemented using the VGG M 1024 CNN [47] adapted for use within the Faster R-CNN framework [259]. Written to support a Python API, the software implementation of Faster R-CNN is founded upon the Caffe deep learning library developed by Jia et al. [150].

The dataset used for evaluating detection and localisation performance of the R-CNN was formed as a union over frames from the FriesianCattle2017 (see Section 3.3.2) and AerialCattle2017 (described prop-

erly in Section 4.2) datasets. This combination of indoor and outdoor imagery yields improved solution generalisation and avoids simplification of the detection task towards a single agricultural environment by introducing background, individual resolution and illumination variation. All original (non-synthesised) images were used from the FriesianCattle2017 dataset. Every 12^{th} frame (or at a rate of 0.5 Hz given 24 fps source footage) for each AerialCattle2017 video was extracted. This combination yielded the final dataset consisting of 1077 images. Two-fold cross validation was performed over this set.

Region predictions from the R-CNN are accepted as true positive provided there is sufficient overlap with a same-class ground truth bounding box via a binary threshold $t_{ov} = 0.5$. Rectangle-rectangle overlap is computed via IoU:

$$ov = \frac{bbox_{GT} \cap bbox_{pred}}{bbox_{GT} \cup bbox_{pred}} \tag{4.1}$$

where $bbox_{GT}$ and $bbox_{pred}$ denote the ground truth and predicted bounding box regions respectively. Categorised detections are subsequently used to compute class precision and recall data. mAP is computed here via the AuC for generated precision-recall curves (see Figure 4.8).

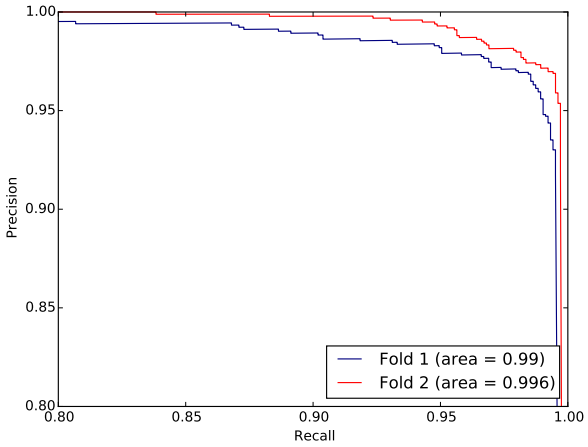


Figure 4.8: ***Precision-Recall Curve for Detection.*** *Detailed section ($x, y \in [0.8, 1]$) of the precision-recall curve for cattle detection and localisation over two-fold cross validation.*

Table 4.1 summarises the two-fold cross-validated performance results, whilst Figure 4.8 illustrates a detail from the corresponding precision-recall curves. This evaluation demonstrates that the task and data tested are very well suited to the employed R-CNN framework. It produces near perfect results of correctly localising cows across both dataset domains and successfully concludes the first component of the iterative identification pipeline illustrated in Figure 4.1.

Figure 4.9 depicts some of the few examples where cattle detection failed. Examined failures consisted of (a): erroneous detections created by the alignment and proximity of multiple cows or (b): false positive detections of partially-visible cattle having no corresponding ground-truth label due to the VOC labelling guidelines on object visibility/occlusion, though, these cases are observably marginal.

Task	mAP (%)		
	Fold 1	Fold 2	Average
Species Detection & Localisation	99.02	99.59	**99.3**

Table 4.1: ***Species Detection Accuracy.*** *mAP values for two-fold cross-validated performance tests for the task of cattle detection and localisation. mAP scores are computed via AuC for generated precision-recall curves shown in Figure 4.8. As demonstrated, the R-CNN cattle detector produces near perfect results for the task of correctly localising cows across the tested datasets.*

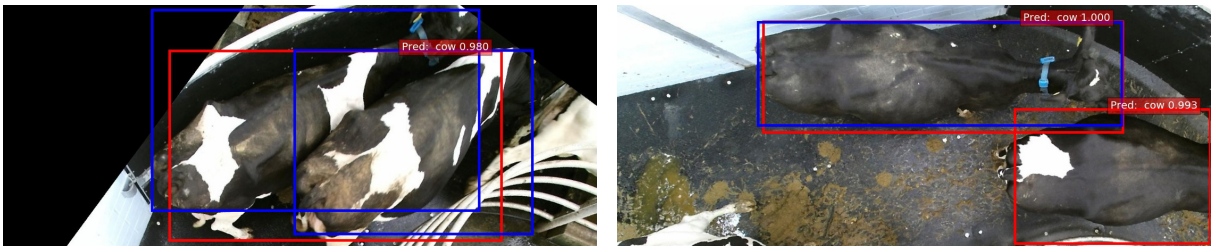


Figure 4.9: ***Species Detection and Localisation Failures.*** *Shown are examples where cattle detection failed due to (left): multi-cow alignment and (right): a controversial erroneous detection due to a lack of ground truth label in adherence with the VOC2012 labelling guidelines [93].*

4.4 LRCN-Based Identification of Tracklets

As has been briefly discussed previously, video – as opposed to a single still image of a scene, event or environment – intrinsically provides an additional (temporal) dimension of information exploitably relevant to individual identification. This section discusses a method for incorporating information from subsequent frames into the identification estimate[6]. The benefit is that complementary information is revealed progressively via new observation viewpoints resulting from camera and target movement, etc. In the specific case of this thesis, observation conditions vary due to movement of the UAV flight platform itself (i.e. varying wind speeds and direction, localisation inaccuracy via GPS), individual cattle movement and more.

In the vast majority of cases, an individual cow can be tracked well within herd videos using the high-speed KCF tracking algorithm [131], given a good initialisation RoI yielded from the predecessor species detector. These *tracklets – sequences of varying individual cattle regions over time – are classified into identities here*. Successful tracking originates from cattle walking relatively slowly – D'Hour et al. find the average Holstein walking speed to be 1.37 m/s over distances of 5.6 and 3.2 km for two trials [69]. And in addition, the segmentation task in an outdoor agricultural setting is somewhat trivial[7] for top-down, aerially-acquired footage. Furthermore, cattle are unlikely to suddenly start walking backwards or change their heading by more than say $\pm 30°$. Thus, if cow_x is present in frame f_i, it is also likely to be present in f_{i+1}[8].

These factors combined with the fact that UAV-captured source footage will exhibit positional and rotational variation due to winds, GPS inaccuracy, etc. contribute towards variation in viewpoint, object configuration and/or scale (for examples, see Figure 4.4). Consequently and importantly, this often reveals the presence of further salient discriminative visual features useful for identification purposes. Such continual assessment of an object's identity over time under varying parameters permits class predictions to be refined and improved iteratively – provided the model can effectively integrate spatio-temporal features – forming much of the motivation for this thesis and its primary contribution.

ANNs featuring LSTM layers [135] fundamentally operate on temporally-based data series, rendering them intrinsically oriented towards the goals of the task here. Whilst alternative forms of recurrent and memory-bearing networks/layers could be employed here, the proven success of LSTM-based architectures over others provides adequate justification of its use here (see Section 2.3.3 for a detailed comparison of alternative architectures). In application to the evaluation of video and image sequences of length n, constituent individual image frames are considered sequentially. For some frame f_i, output from a LSTM layer is fed as input to the subsequent iteration for frame f_{i+1}. In the case of the task here,

[6] In operation on identifying individuals in video data in this section, only videos acquired via the use of a UAV and as part of the AerialCattle2017 dataset (see Section 4.2) are used here. Other datasets FriesianCattle2015 and FriesianCattle2017 are comprised of singular image frames only.

[7] Black and white cow targets with a typically green grass background.

[8] Given a sufficiently frequent frame-rate in source footage.

following processing of frame f_n, a final class-prediction vector is yielded for the entire input sequence via a fully connected layer employing softmax activation.

Convolutional feature representations of input frames f_i are extracted prior to input into the LSTM layer using a pre-trained Inception V3 CNN [303]. The network is trained on single images of individual identities and feature maps are extracted just before the final output classifier. Such a combination of a CNN and LSTM layer(s) – first introduced by Donahue et al. [73] – is referred to as a LRCN architecture and is employed here for the task of spatio-temporal individual identification. Figure 4.10 illustrates the standard LRCN pipeline used. It is proposed that this core identification pipeline can then be easily integrated into surrounding components of the complete video processing architecture as illustrated in Figure 4.1.

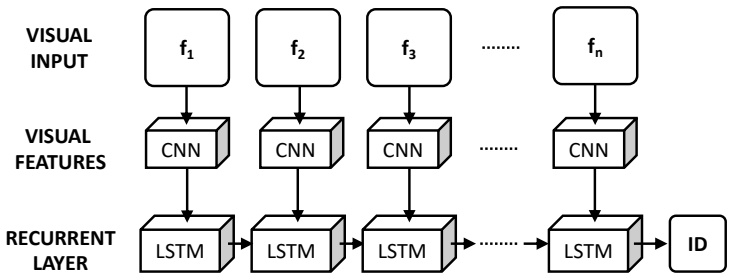


Figure 4.10: ***Recurrent Convolutional Architecture.*** *Unrolled identification refinement pipeline for an input video based on the LRCN architecture [73]. Visual features for input video frames $\{f_1, f_2, ..., f_n\}$ are extracted via an Inception V3 CNN [303] trained to identify individuals from single images. Feature maps are extracted from the penultimate layer for input into a single LSTM layer with* 256 *units ultimately yielding an identity prediction for the entire image sequence.*

4.4.1 Experiments

The dataset used for this task originates solely from the outdoor video dataset (AerialCattle2017) and consists of 46,430 cropped image frames and 23 individual cows over 160 videos. Tracklet instances were sequentially split into 40-frame long spatio-temporal streams. Performing this stage for the entire data corpus resulted in 1064 labelled streams, each containing a single individual over time. This data was then partitioned[9] into a ratio of 9:1 (that is into 957 and 107 streams), for training and testing respectively. Note that consequently, training and testing instances may originate from the same original video and thus, may be visually similar.

To perform identification refinement, the Inception V3 network [303], an extension of GoogLeNet [302], was fine-tuned on the 23-class training dataset. More specifically, each frame from the 957 training streams was used to fine-tune Inception weights originally trained for the ILSVRC [270]. Subsequently, all frames from each training stream were passed through the re-trained network. The 2048-wide vector yielded at the 'pool3-layer' was captured in each case. Convolutional frame-wise representations for each stream were then used to train a single LSTM layer comprised of 256 cells followed by a fully connected layer with $|classes| = 23$ neurons and softmax activation. This architecture was selected following experiments with alternative network architectures (e.g. multiple stacked LSTM layers, varying the number of cells, etc.) resulting in worse final identification performance on the validation set overall.

Figure 4.11 illustrates improving prediction accuracies on the training and testing datasets over 1,000 epochs, with identification results given in Table 4.2. For each prediction, an ordered vector of size $|classes| = 23$ is produced with individual-wise class confidences $\in [0, 1]$ subject to the condition that the vector summates to 1. The predicted class label is taken to be the index of the maximal value in

[9] Data partitioned using the `train_test_split` SciKit-Learn Python machine learning library function: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html.

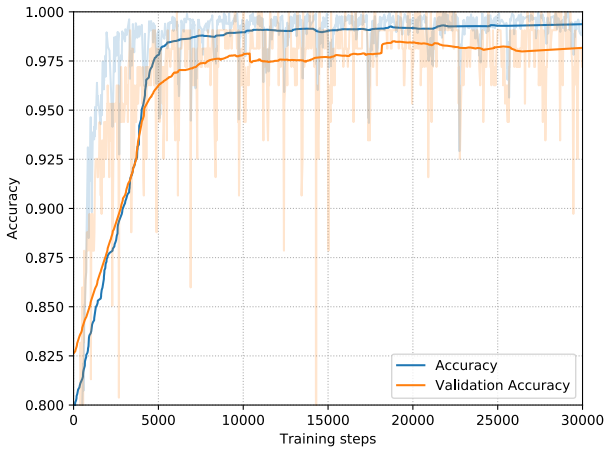


Figure 4.11: ***Training Towards Individual Identification.*** *Detailed section of prediction accuracy on the training and testing/validation set for identification at different stages of LSTM training over 1,000 epochs[10] on the training and testing datasets consisting of 957 and 107 streams, respectively.*

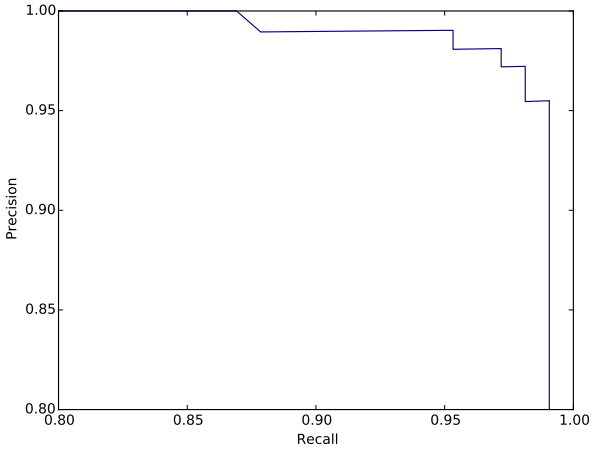


Figure 4.12: ***Precision-Recall Curve for Video Identification Evaluation.*** *Detailed precision-recall curve* $(x, y \in [0.8, 1])$ *for LRCN identification on retained testing data consisting of 23 possible cow classes for 107 testing video streams.*

that vector. A prediction is then considered a true positive if the predicted and ground truth class labels match.

Task	Accuracy (%)	
	Training	Testing
LRCN Individual Identification	99.79	**98.13**

Table 4.2: ***Video Individual Identification Accuracy.*** *Classification accuracies of the LRCN setup after training for 1,000 epochs on the 1064 instance-strong video stream dataset (training set: 957 instances and testing: 107) containing image sequences of individual Holstein Friesians. Each instance consists of 40-frame image sequences comprising* ~ 1.5*s of video.*

Figure 4.12 shows a detailed section of the precision-recall curve for the iterative identification task, whilst some examples of false positive classifications are given in Figure 4.13. For erroneous predictions, the individuals referred to by predicted and ground truth class labels were visually similar. The quantity (number of white, black or brown patches), position and shape/structure of coat pattern features were found to bare strong resemblance across mistaken labels.

(a) $ID_{PRED} = 3$, $ID_{GT} = 10$. *(b)* $ID_{PRED} = 11$, $ID_{GT} = 14$.

Figure 4.13: ***Video Identification Failures.*** *Examples of false positive classifications from LRCN identification on the AerialCattle2017 dataset. In all failure cases, dorsal features are observably visually similar in structure and positional distribution on the body.*

[10] Signals smoothed using the Savitzky-Golay filter [275].

4.4.2 Quantification of Iterative Identification Advantage

In this section, the primary intention is to quantitatively and qualitatively reveal how a paradigm of performing identification over multiple iterations is beneficial as opposed to the traditional approach of evaluating a single frame. Towards this goal, figures/graphs have been selected to illustrate the internal operational mechanics of the LRCN model towards uncovering some of its inherent "black box"-like properties; its behaviour to relevant inputs.

To begin, what is meant here by model confidence and identification satisfaction is given to remind the reader. Output of neural network models with a final softmax function-activated layer is a $|classes|$-dimensional vector K where:

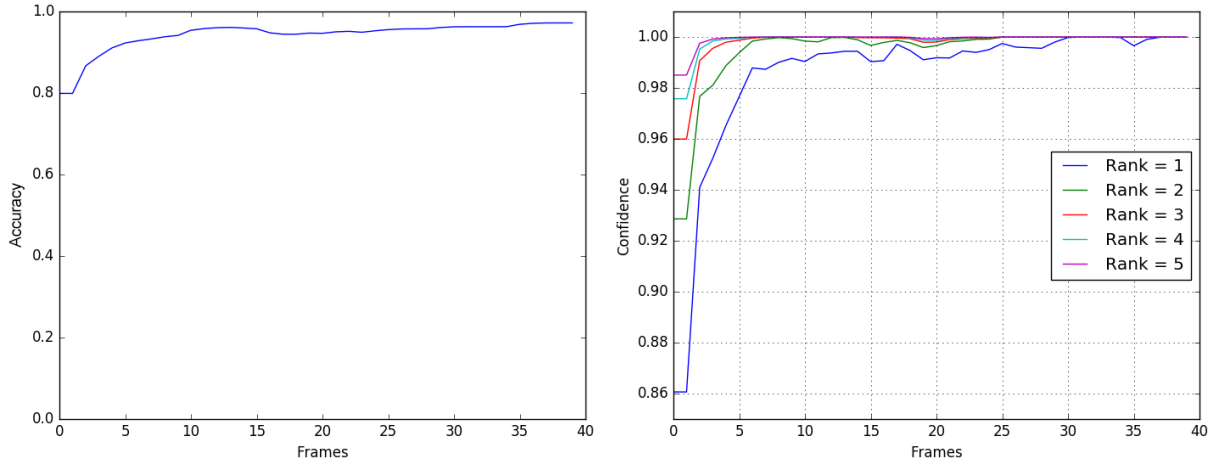
$$K = \{k_0, k_1, ..., k_n\} \tag{4.2}$$

subject to:

$$\forall k_i \in K,\ k_i \in (0,1) \ \text{ and } \ \sum_{i=0}^{n} k_i = 1, \tag{4.3}$$

where $n = |classes|$ and $k_i \in \mathbb{R}^+$. Each value k_i is interpreted here to be the model's inferred confidence in the class c_i for some input I, where $classes = \{c_0, c_i, ..., c_n\}$. If a particular probability value k_j satisfies a defined threshold $k_j \geq \alpha,\ \alpha \in \mathbb{R}^+$, identification confidence in the class c_j is considered satisfactory.

As a first step towards justifying the need for a multi-iteration paradigm at all, model accuracy was collected across all 107 testing instances utilised in the previous experiment (see section 4.4.1). This is exemplified in Figure 4.14a, where a clear trend visibly indicates that overall accuracy increases versus exposure to multiple subsequent images/frames and corresponding feature vectors – equally applying to model confidence in some category (see Figure 4.14b). This demonstrates the success of the trained LRCN model beneficially integrating spatial convolutional features from multiple sources. Whilst the trend line itself is not perfectly linear and/or uniform, an increased testing sample size would solidify trend appearances. These results then serve as a proof-of-concept for future experiments upon larger datasets consisting of more instances and more classes. The core takeaway of this analysis is that: *individual identification demonstrably benefits from exposure to a progressive stream of subject-wise visual information.*



(a) Rank-1 ***accuracy*** *vs. frames.*

(b) 5-rank model ***confidence*** *vs. frames.*

Figure 4.14: ***LRCN Model Performance vs. Progressive Frame Exposure.*** *Graph (a): model accuracy versus exposure to subsequent complementing frames. At each frame iteration j, the maximal element of the yielded confidence vector $argmax(K_j) = c_j$ is taken to be the predicted label. If the prediction matches the ground truth $c_j = c_{GT}$ for that testing instance, a correct classification is attributed to that instance at frame j. This is repeated over all* 107 *instances for all* 40 *frames to produce an accuracy metric versus frames. Graph (b) illustrates 5-rank model confidence (see equation 4.2) in some class label versus frames.*

Importantly however, in the vast majority of testing cases, evaluation of the first frame in the sequence provides sufficient evidence for the ground truth identity. As can be seen in Figure 4.14a, $86/107 = 80.37\%$ are actually predicted correctly. The traditional approach of evaluating a single frame therefore, is demonstrably strong, highlighting the success of the predecessor species detection component in providing appropriate individual-wise RoIs that are identified here. Upon exposure to new frames undergoing variance, discriminative features are revealed progressively, improving accuracy overall. This additionally suggests that object regions are well-associated and maintained over image sequences from the KCF tracking algorithm [131].

Figure 4.15 illustrates the number of frames required to identify the individual in question to confidence level α. The takeaway being that for complete model confidence ($\alpha \approx 1, \sim 100\%$[11]) in some correct or incorrect label, the LRCN must be exposed to multiple iterations featuring visual variation. However, the single-frame paradigm is mostly sufficient when this constraint is slightly relaxed (e.g. $\alpha = \{0.9, 0.95\}$). Finally, it is visible that there are some testing instances – irrespective of model confidence requirements – that require the multi-iteration identification approach described in this chapter.

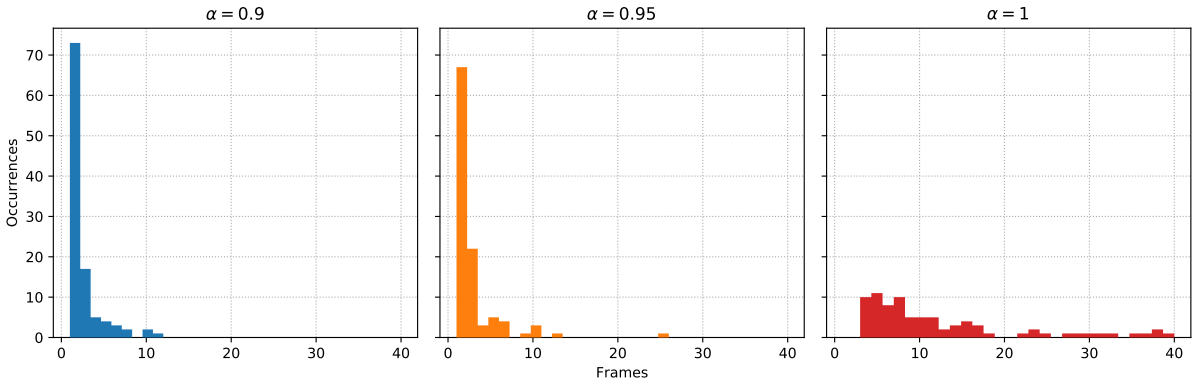


Figure 4.15: ***Frames Required for Identity Confidence Threshold Satisfaction.*** *Histogram of the number of frames/images required incrementally in order to satisfactorily identify a particular testing individual-wise video stream to confidence levels* $\alpha = \{0.9, 0.95, 1\}$.

Hand-picked testing instances of particular interest are depicted in Figure 4.16 illustrating occasions where multiple frames were required to result in the correct identity prediction, whilst Figure 4.17 illustrates similar graphs for the only three failures out of all 107 testing instances. With respect to multi-iteration successes (Figure 4.16), there appear to be several classifications of graph characteristics that relate to: $(a), (c), (e)$ a complete switch in prediction (to the correct label) at some event e, $(b), (f)$ fluctuations in the correct label and (d) complete identity prediction confusion until some event e. This is similarly the case for identification failure cases shown in Figure 4.17. The important takeaway of both examples of successes and failures is that the rate of change in model confidence versus time is relatively high. This corresponds to the visual information revealed at event e or frame f_i providing an abrupt spike in the knowledge required for the model to solidify confidence in a particular identity/class. Put differently, the LRCN-based identification pipeline seemingly does not integrate spatio-temporal features as gradually as the original Figure 4.14 illustrating average confidence versus frames might suggest. It is rather the observation of one particular disambiguating frame and its constituent discriminative spatial features that appears to solidify model confidence in some category.

To finish, a short experiment in which the ordering of frames in testing instances was randomised observed a negligible difference in attainable prediction accuracy. This corresponds to the fact that temporal features themselves are unimportant in this experimental context, it is rather the observation of particular spatial features occurring at some stage that prevails and LSTM layers offer a convenient and demonstrable ability in meaningfully combining this evidence.

[11] Absolute confidence $\alpha = 1$ is not possible due to softmax enforcing non-zero elements $\forall k_i \in K$ – see equation 4.3.

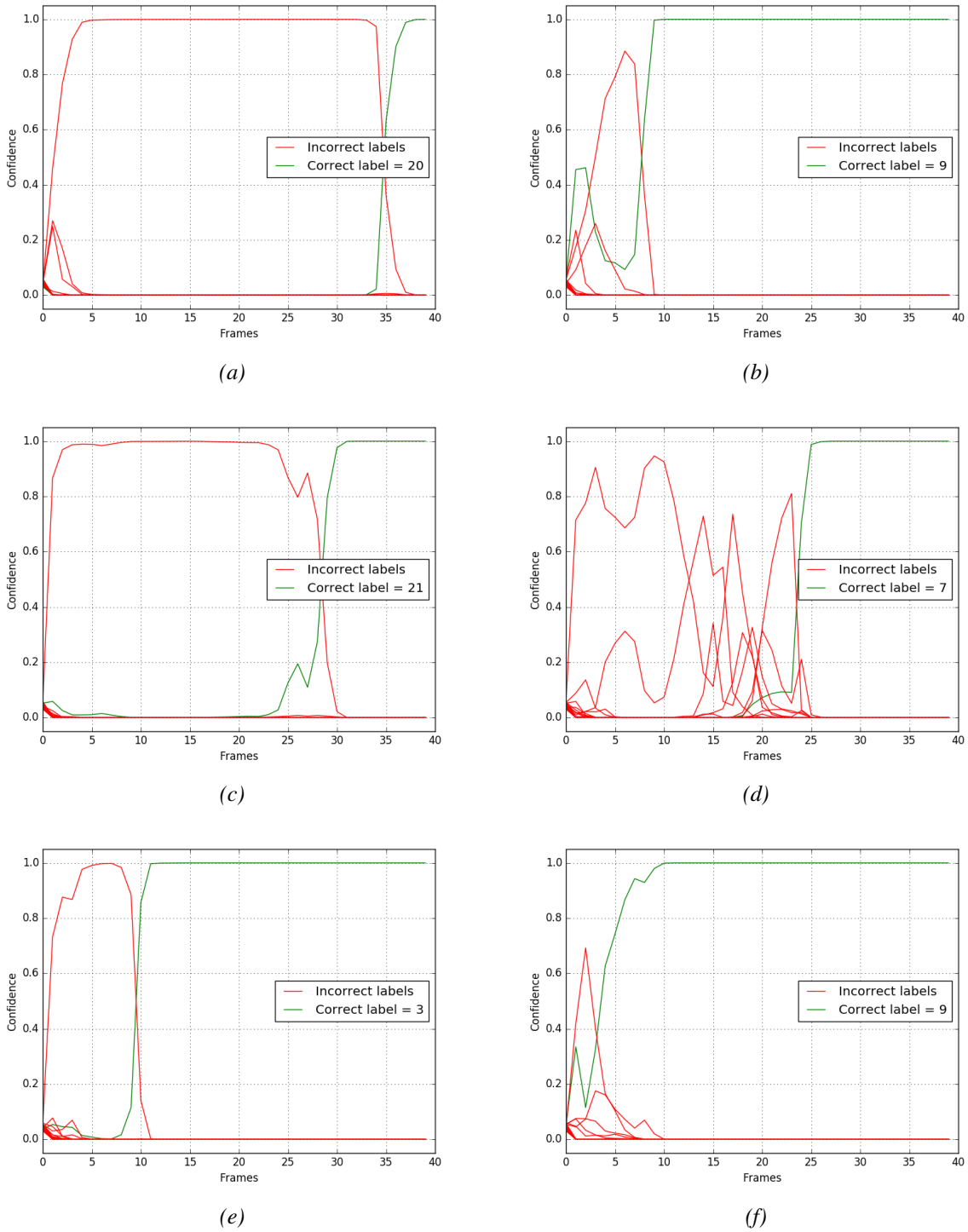


Figure 4.16: ***LRCN* Model Confidence Versus Time/Frames.** *Hand-picked interesting examples out of* 107 *testing instances where multiple frames were required to ultimately yield the correct identity prediction – some particular discriminative feature for that category becomes visible at frame* f_i *(e.g. frame number* 8 *for graph* (e)*). For each example, model confidence vs. exposure to subsequent complementing frames in the* 40 *frame-long image sequence is shown, where green and red denote confidence in the correct ground truth class* c_{GT} *and all other incorrect labels, respectively. Resulting from softmax enforcing confidence summation to* 1 *(see equation 4.3), symmetry is visible for some instances within particular frame windows – graph* (c) *illustrates this occurrence well. The graphs in* (a)*,* (b) *demonstrate that employing a simpler mechanism of a voting scheme on frames considered individually would fail in some circumstances.*

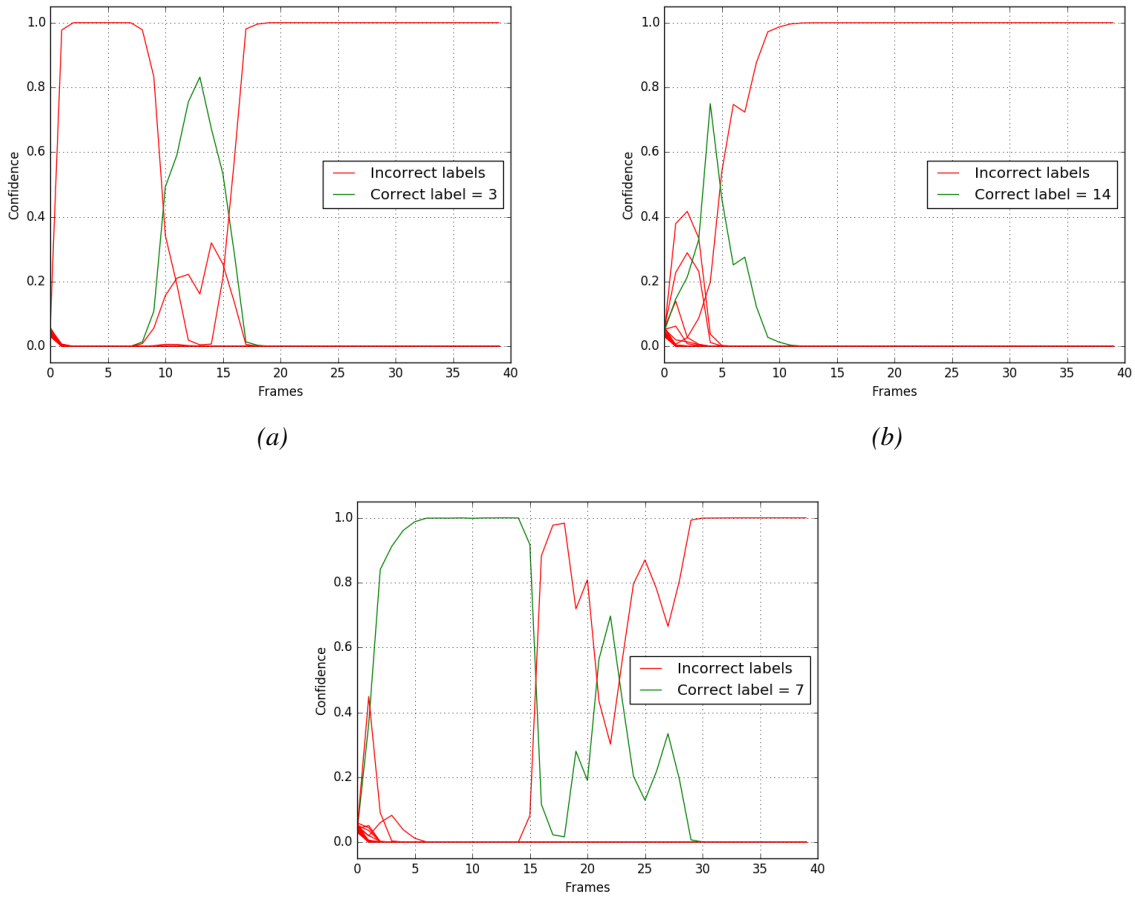


Figure 4.17: ***LRCN Identification Failure Cases.*** *Example graphs of model confidence versus time/frames for failed instances, where the incorrect label was ultimately predicted. These failures were found to result from intra-category visual similarities from viewpoint origination (see Figure 4.13), and a lack of variation across the* 40 *frames of the respective spatio-temporal streams.*

4.5 Chapter Conclusion

This chapter proposes a novel architecture successfully realising automated passive iterative individual Holstein Friesian cattle identification in an agriculturally-relevant outdoor field environment. As part of this, pipeline components solving species-wide detection and localisation as well as individual tracking are successfully established in providing strong initialisation to later components. This pipeline extends the work completed in the previous chapter operating within a single-frame, single-evaluation paradigm to demonstrate that identification directly benefits in prediction accuracy from an iterative process. More specifically, the exposure of multiple iterations (frames) to the identification model proposed here under varying observation conditions and parameters is proven advantageous as a result of the employed architecture and pipeline successfully extracting and classifying salient spatio-temporal features for subject target individuals.

This being said, throughout the identification process conducted here, the UAV was flown entirely manually and merely captured data. It was not informed by the identification process and therefore can be regarded as a completely passive scenario; captured video footage was analysed by the LRCN-based pipeline in an offline setting. Whilst it has been shown that this can prove sufficient in the presented scenarios, work in the subsequent chapter will explore actively involving an agent within the identification process towards improved accuracy and efficiency in a comprehensive simulation environment.

Supplementary Information

Publication associated with this chapter:

1. W. Andrew, C. Greatwood, T. Burghardt. Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning. In *IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 2850-2859, 2017. https://doi.org/10.1109/ICCVW.2017.336.

Accompanying published datasets available at:

- **FriesianCattle2017**:
  https://data.bris.uk/data/dataset/2yizcfbkuv4352pzc32n54371r
- **AerialCattle2017**:
  https://data.bris.ac.uk/data/dataset/3owflku95bxsx24643cybxu3qh

Chapter 5

Simulated Active Multi-Frame Identification

5.1 Chapter Overview

This chapter introduces active vision to the identification process. To this point, individual identification of cattle has occurred in an entirely passive setting; firstly, by evaluating a single captured image and second, evaluating multiple temporally-variant frames of a subject. Put differently, the agent has had *no* influence on the identification process up until this point. Accordingly, this chapter proposes a method for extracting and observing individual-specific viewpoint trajectories that satisfactorily identify the subject as quickly as possible. This method is comprised of a multi-network DNN pipeline that infers identity from sequentially presented imagery. The choice of imagery (the object viewpoint) is taken care of by the same architecture actively seeking new viewpoints that reveal salient information about that particular subject whilst minimising the distant travelled to observe said viewpoints (as is costly on-board a UAV-based agent). The validity of such an approach is demonstrated in realistic three dimensional simulations of a small population size with manually generated textures designed to contrive difficult identification cases.

This chapter is organised, first and foremost into a brief introduction (Section 5.2) and next, a formalisation of the mathematical foundations this chapter utilises throughout (see Section 5.3). Thirdly, implementation choices and details are given for individual components together forming the proposed active identification pipeline in Section 5.4. This is followed by appropriate experiments, results and accompanying findings (Section 5.5), finally finishing with concluding remarks in Section 5.6.

5.2 Introduction

This chapter presents a solution for an agent actively and visually identifying a single subject individual from a larger population. The agent is considered here to be able to move freely about three dimensional space and can point its camera in any orientation, thus having 6 DoF: x, y, z and camera roll (ϕ), pitch (θ) and yaw (ψ) – much like a real-world UAV. The intention is to perform identification by integrating information from multiple frames or iterations, where each frame is captured from a new viewpoint. New viewpoints are chosen by the agent to maximise the likelihood of identifying the individual in question. The choice of observations adhere to minimising the distance travelled by the agent and minimising the number of observations. In this form, the agent performs active identification of the individual whilst attempting to minimise the cost of the solution. An important question that arises from this problem formulation revolves around termination criteria. That is, how does the agent deem a target individual to be identified? Here, this criteria is set to be the identification model being confident in some identity to belief level α in some identity class. Solving this form of identification agency, this chapter proposes a pipeline comprised of several deep architectures that are trained individually to together form the complete framework. As a result, examples of what constitutes a good solution are determined and presented to the architecture performing active identification.

5.3 Mathematical Formalisation

5.3.1 Representations

Within this chapter and its corresponding simulation environment, the world is modelled as a continuous three dimensional environment E. Note that, wherever applicable, a right-handed coordinate system/frame is employed throughout and time is discretised; $t_i \in \{t_0, t_1, ..., t_e\}$.

Agent

The agent G takes continuous three dimensional coordinates in E:

$$\vec{G} = (G_x, G_y, G_z), \tag{5.1}$$

where coordinate $(0,0,0)$ is the origin of E and $\vec{G}$ defines the centre of mass of the agent/UAV model. Alongside its position is the current agent yaw rotation value expressed as a heading $G_\psi \in [-\pi, \pi]$ aligned with the quadrants of $atan2(\cdot)$. The reference frame of the agent is aligned with a Front Left Up (FLU) axis (see Figure 5.1 for illustrative purposes). Additionally, the camera gimbal system is assumed to be rotationally independent and its orientation or "look at" point defined by $C = (C_\theta, C_\psi)$ denoting camera pitch and yaw angles, respectively, with:

$$\begin{aligned} C_\theta &\in [0, -\pi], \\ C_\psi &\in [-\pi, \pi]. \end{aligned} \tag{5.2}$$

Finally, to maintain memory of belief in identity of an encountered individual, the agent maintains a vector:

$$\begin{gathered} K = (k_0, k_1, k_2, ..., k_n), \\ \text{where } \; n = |R|, \; k_i \in (0,1), \; k_i \in \mathbb{R}^+, \; \sum_{i=0}^{n} k_i = 1, \end{gathered} \tag{5.3}$$

encoding per-identity confidence values as yielded from a softmax function. Note that the set $R = \{r_0, r_1, ..., r_n\}$ defines the set of targets described in the following section. Accordingly, the state of the agent for some time-step t is therefore given by the following tuple with components summarised in the following Table:

$$S^t = (\vec{G}^t, G^t_\psi, C^t, K^t). \tag{5.4}$$

Symbol	Description
$\vec{G}$	Global 3D agent position vector in the world/environment.
G_ψ	Agent yaw heading.
C	Camera gimbal parameters C_θ, C_ψ denoting camera pitch and yaw angles respectively.
K	Memorisation of confidence in observed identities.

Note that camera gimbal roll parameter C_ϕ is ignored in the formulation here and indeed in this chapter as well as the remainder of this thesis. The camera is assumed to always be level with the horizon – made possible in reality via the use of a 3-axis gimbal system – such that image variance from rotation is not introduced. This is especially important since convolutional architectures are intrinsically not invariant to object rotation [330].

Targets

As is common in computer vision applications, targets representing individual cattle are defined here in an abstract form; as a three dimensional cuboid. A target $r \in R$ is thus, spatially defined by three orthogonal lengths:

$$r = r_l \times r_w \times r_h, \tag{5.5}$$

where l, w, h denote cuboid length, width and height, respectively, with values chosen as:

$$\begin{aligned} r_l &= 2.7\text{m}, \\ r_w &= 1.03\text{m}, \\ r_h &= 1.65\text{m}, \end{aligned} \tag{5.6}$$

as approximately for an meanly-sized, middle-aged female Holstein Friesian [312]. Targets take a global position $\in E$:

$$\vec{A} = (r_x, r_y, r_z), \tag{5.7}$$

centred within the target itself, alongside a yaw heading value r_θ. To complete the definition, each individual target in an experiment is assigned an exclusive unique identifier:

$$\begin{aligned} r_{ID} &\in [0, |R| - 1], \\ r_{ID} &\in \mathbb{Z}, \end{aligned} \tag{5.8}$$

in correspondence with synthetic textures generated for the simulation environment (for examples, see Figures 5.12 and 5.18).

5.3.2 Active Identification

With respect to active identification, for some visual observation I^i made by the agent with state S^i of a target $r_j \in R$ at time-step i, we want to fulfil a new state configuration S^{i+1} to conduct a new observation I^{i+1} that maximises the agent's confidence $k_j \in K^{i+1}$ in the target r_j (currently being observed). Put differently, the intention is that the new observation I^{i+1} made from new configuration S^{i+1} – comprised of a new agent position command together with new camera pitch and yaw angles – maximises the likelihood of the model identifying the current target r_j whilst minimising solution cost overall (cost function design is discussed later in Section 5.4.5). For the target r_j to qualify as being deemed to be satisfactorily identified, a corresponding confidence value $k_j \in (0,1)$ (yielded from the active identification model) must exceed a defined threshold $\alpha \in (0,1), \alpha \in \mathbb{R}^+$. New, subsequent agent states are computed via a function mapping:

$$f(S^i) \longrightarrow S^{i+1}, \tag{5.9}$$

or more explicitly;

$$f(\vec{G}^i, G^i_\psi, C^i, K^i) \longrightarrow (\vec{G^{i+1}}, G^{i+1}_\psi, C^{i+1}, K^{i+1}), \tag{5.10}$$

and are target specific with respect to an individual's dorsal coat pattern and markings. The goal therefore, is to learn the function $f(\cdot)$ such that it can be deployed for *online* and *active* individual identification when it is required as such.

Within the function $f(\cdot)$ from equation 5.10 mapping from one state to another to be fulfilled, simplifications can be made towards expression of the problem under more manageable terms. Firstly, the agent's yaw can be ignored since it is maintained from iteration to iteration to be aligned with a manually-defined reference frame. Specifically within this chapter operating in simulation, the front of the aircraft is kept aligned with $+y$ throughout experiments. Instead the camera rotates in z or yaw as required. Second, goal camera parameters $C^{i+1} = (C_\theta, C_\psi)$ are also ignored here as their determination is trivial

in keeping the image frame centred about the target currently under identification investigation (full solution described in Section 5.4.1). Finally, the consequent confidence vector K^{i+1} can be ignored as it is a passive component of the agent's state which is just observed and updated per iteration. The result of these simplifications reduce the original problem to mapping the agent's current state to a new goal position:

$$f(\vec{G}^i, G^i_\psi, C^i, K^i) \longrightarrow \vec{G^{i+1}}, \tag{5.11}$$

that *maximally improves the confidence estimation in the current individual to a satisfactory level* α *with minimal overall cost*.

5.4 Implementation

A summary of the implemented active identification pipeline is visually depicted in Figure 5.2. In order to accomplish the simplified learning of function $f(\cdot)$ established previously in equation 5.11, the mapping itself is solved by a "divide and conquer" approach. The justification being that the parameter search landscape remains vast, complex and ultimately computationally intractable despite simplification of the core function approximation. The generation of appropriate training data itself is therefore difficult, as intuitively, what is a good active identification solution? Instead, the problem is broken down into core components described as follows.

Firstly, in order to ever yield a new goal agent position $\vec{G^{i+1}}$ that observes an unseen target viewpoint, an understanding of where the target in question is positionally situated must be formed:

$$\xi(I^i, C^i) \longrightarrow \vec{D}^i. \tag{5.12}$$

Function $\xi(\cdot,\cdot)$ takes the current image (containing target r_j) and camera parameters as input to produce the three-dimensional position vector $\vec{D}$ between the agent and the target's centre of mass in the agent's reference frame. Following transformation and in summation with the agent's current position $\vec{G}$, this forms a global position estimate of $\vec{A} = (r_x, r_y, r_z)$ for the target r_j that permits new viewpoints to be fulfilled. Here, function $\xi(\cdot,\cdot)$ is approximated by a deep network performing 3D regression as described properly in Section 5.4.2, whilst Figure 5.1 illustrates the estimation concept.

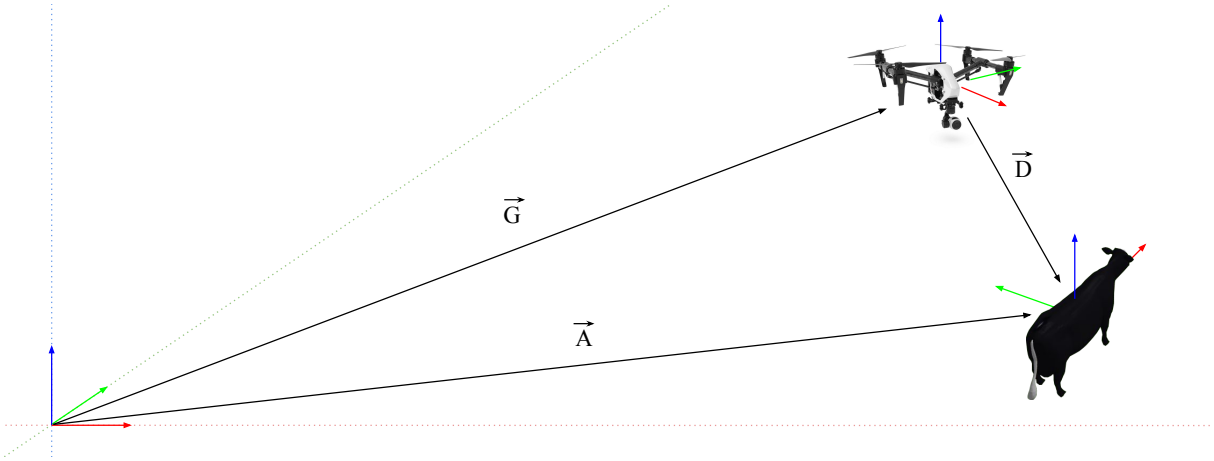


Figure 5.1: ***3D Target Displacement Estimation.*** *Illustration*[1] *of the Agent-Target Estimation (ATE) problem formulation estimating the 3D displacement between the agent and the current target* $\vec{D}$*. The agent G and target* r_i *have FLU-aligned axes. The position of* r_i *is approximated via* $\vec{A} = \vec{G} + \mathbf{T}_W(\vec{D})$*, where* $\mathbf{T}_W(.)$ *transforms an input vector into the world frame given knowledge of the agent's camera attitude* C_θ, C_ψ*.*

[1] UAV image credit: DJI

Second, to complete potential fulfilment of a new observation viewpoint, a determination or estimation of the target's pose relative to the agent must take place. Put differently, an estimation of what area or segment of the target is currently being observed must be formulated in order to inform the identification process. In order to achieve this and abstract away from complex formulations, the problem is cast as a discrete classification problem here, as described properly in Section 5.4.3.

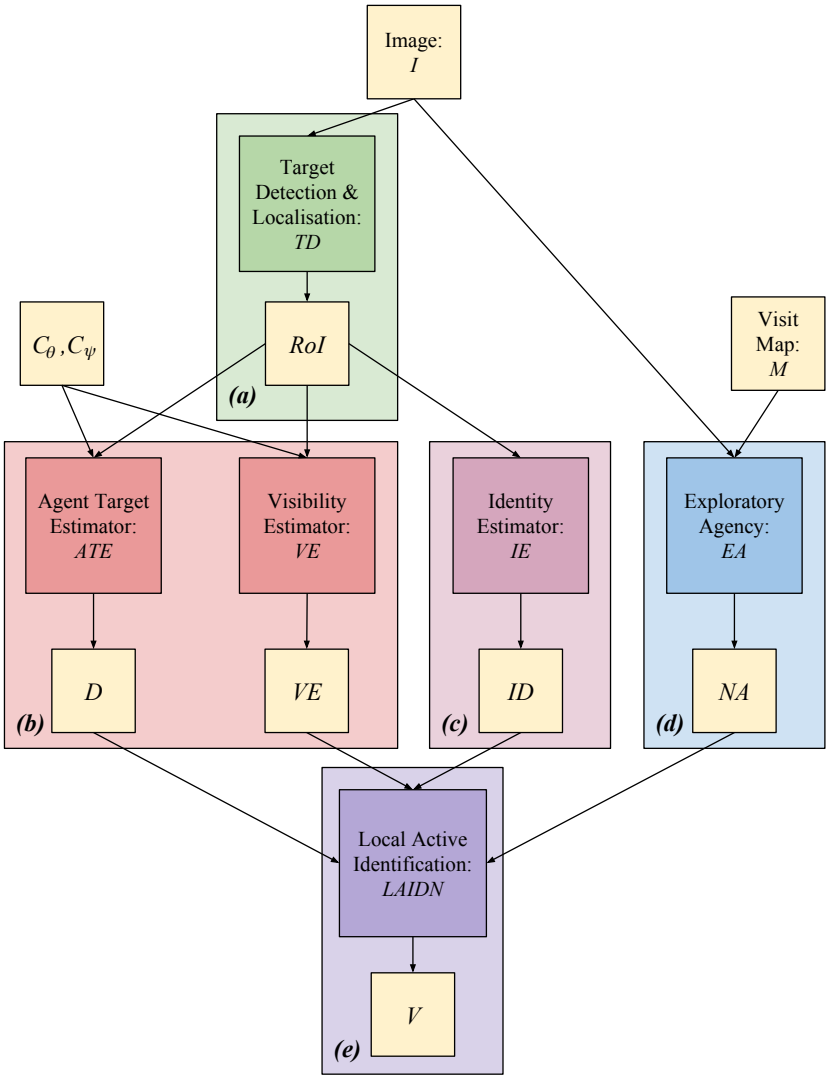
Now that a potential new observation viewpoint can actually be fulfilled, motivation of such an undertaking is informed by a *third* component estimating target identity r_{ID}. This task is intrinsically a classification task (described properly in Section 5.4.4) yielding identity confidence vector K. If this process does not satisfactorily identify the target in question (i.e. yielding model confidence in excess of a threshold α), a new observation is needed under observation parameter variation.

The *fourth* and final component is utilised when a single image was not sufficient in identifying the current target r_j visually. Its goal is to produce a new agent-centric position $\vec{G^{i+1}}$ that maximises the likelihood of identifying r_j whilst maintaining minimal overall solution cost (see Section 5.4.5 for full implementation details). This is accomplished in utilising the outputs of predecessor networks as inputs to this network component by providing contextual information regarding the current identification problem. The full network architecture and its constituent individual components are given in Figure 5.2.

Within this chapter, the area in which local and active identification is deemed to occur is defined spatially by:

$$\begin{aligned} &l = 4\text{m}, \\ &w = 4\text{m}, \\ &h_l = 5\text{m},\ h_u = 7\text{m}, \\ &\text{with}\ \ x \in [-l, l],\ y \in [-w, w],\ z \in [h_l, h_u]\ \ \text{and}\ \ x, y, z \in \mathbb{R}, \end{aligned} \tag{5.13}$$

for a target model placed at the origin, where x, y, z comprise possible 3D coordinates within the operational bounding box. These values (defined in metres) are chosen to mimic and optimise the constraints imposed by real experimentation. Chosen with the size of real cattle targets in mind (as defined in equation 5.6), in conjunction with possible exploratory agency grid cell size, application and domain-specific requirements dictate suitable choices of parameters given above in equation 5.13. Minimum possible height value $h_l = 5\text{m}$ is the direct result of previous Chapter 4 in which, cattle eventually displayed comfort towards a UAV operating at that altitude. Note that this chapter operates within a realistic simulation environment only. For full implementation details, refer to Appendix A.



Group	Network Name	Notation	Section	Input(s)	Output(s)
(a)	Target Detector	TD	5.4.1	Image I	RoI
(b)	Agent-Target Estimator	ATE	5.4.2	RoI, C_θ, C_ψ	$\vec{D}$
(b)	Visibility Estimator	VE	5.4.3	RoI, C_θ, C_ψ	VE^0_{out}, VE^1_{out}
(c)	Identity Estimator	IE	5.4.4	RoI	ID
(d)	Exploratory Agency	EA	5.4.5	Image I, Visit map M	$NA = a,\ a \in A$
(e)	Local Active Identification	LAIDN	5.4.5	$LAIDN_{in}$	$V = \vec{LAIDN}_{out}$

Figure 5.2: ***Complete Active Identification Pipeline.*** *Illustration of the complete pipeline for the performance of active and online identification. Assuming the camera is pointed at an individual to be identified, processing begins with sensory sampling; the agent's current camera gimbal parameters pitch and yaw, camera image and environment visitation map are retrieved (more on exploratory agency in later Chapter 6). The corresponding image is given as input into* ***(a)****: performing target detection and localisation yielding an individual-wise bounding box or RoI. This smaller image in conjunction with camera rotation parameters are given as input into* ***(b)****: networks estimating the the three dimensional vector between the agent and target as well as which segment of the target is currently visible. The same RoI is also given as input into* ***(c)****: which concatenates this RoI with images of the current individual from previous iterations to perform iterative identification on a sequence of images. The original image I is also used to* ***(d)****: yield global exploratory decisions that are fulfilled after the individual has been identified (again, more on this in later Chapter 6). Finally,* ***(e)****: outputs from predecessor networks are concatenated to form $LAIDN_{in}$ input into a MLP trained to produce a three-dimensional vector V the agent will displace itself by that identifies the individual in question with minimal cost (a combination of flight time and number of iterations). Identification is halted once an identity estimate* ***(c)*** *is confident in some identity above threshold α, at which point, exploratory agency prevails to discover new individuals in the environment to be identified.*

5.4.1 Target Detection and Localisation (TD)

Target (simulated cattle) detection is performed by state-of-the-art and real-time detector YOLO v2[2] [257]. The goal of this stage is two-fold: *(a)* detecting whether target(s) are currently present in the image and *(b)* localising an image RoI or bounding box to each detection. The primary takeaway is that in complete operational success, this stage yields sub-images containing a particular single target only, whilst minimising the quantity and resulting proportion of background pixels. Successor networks then take these regions as target-centric image inputs (refer to Figure 5.2 for the full operational pipeline). Note that image regions are resized non-proportionally to a fixed array size $224 \times 224 \times 3$ for divisibility by 32 and input into subsequent networks mandating static input tensor size.

Difficulties in this work-flow arise when some image I contains multiple individuals, therefore potentially yielding multiple detections. The question is: which detection should be selected and provided as input to subsequent networks or, which individual is currently being identified? Since targets are kept central in the original image via the "look at" functionality described in Section 5.4.1 to maximise target pixel coverage likelihood, the central-most satisfactorily confident detection is chosen.

Viewpoint-Dependent Training Data Generation

To generate appropriately labelled training data that is subsequently used to train the YOLO architecture (see Figure 5.3 for an example), a random cow model $r_i \in R$ is placed at the origin and a random Agent/UAV position is generated:

$$\begin{aligned} &\vec{G} = (G_x, G_y, G_z), \\ &\text{where } \; G_x \in [-w, w], \; G_y \in [-l, l], \; G_z \in [h_l, h_u] \; \text{ and } \; G_x, G_y, G_z \in \mathbb{R}, \end{aligned} \tag{5.14}$$

where w, l, h_l, h_u defined in equation 5.13 denote the boundaries of the area for local active identification. Appropriate gimbal pitch C_θ and yaw C_ψ values for that position are computed such that the model centre of mass given by the coordinate $(0, 0, \frac{cow_h}{2})$ is intersected by the camera optical axis using equations 5.15 and 5.16 respectively. The result being that the cow model is perfectly central in the synthetic UAV image as depicted in Figure 5.3.

$$C_\theta = \tan^{-1}\left(\frac{\sqrt{|G_x|^2 + |G_y|^2}}{G_z - \frac{cow_h}{2}}\right) \tag{5.15}$$

$$C_\psi = \text{atan2}(-G_y, -G_x) \tag{5.16}$$

Next, eight 3-dimensional coordinates were manually generated and verified such that they enclose the target cow model spatially – defining a three-dimensional object bounding box (see Figure 5.3a). All that remains is to generate the ground truth 2D image bounding box signifying an object annotation or RoI. This is achieved by appropriately projecting 3-dimensional bounding box coordinates onto the image plane. To obtain pixel coordinates (x, y) for a 3-dimensional point in the world coordinate system $P = (X, Y, Z)$, the point P must be expressed in the camera coordinate system yielding P_c. In order to achieve this, the world-to-camera homogeneous transformation is utilised:

$$P_c = \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{5.17}$$

[2]Whilst there exists a newer, improved iteration of the detector in the form of version 3 [258], at the time of writing there are no suitable interfacing libraries/API's between YOLO (founded in DarkNet) and TensorFlow, as utilised here. This isn't the case for earlier versions of YOLO and hence; version 2 and DarkFlow are used here. Darkflow repository: https://github.com/thtrieu/darkflow

where $r_{ij},\ i,j \in [1,3]$ and $t_k,\ k \in [1,3]$ denote world-to-camera rotation and translation parameters, respectively. Rotational parameters r_{ij} are determined by Euler angles for the axis sequence ZYZ with values $\frac{\pi}{2}, -C_\theta, -C_\psi$ respectively. This camera reference frame relative point is then computed in uvw space via:

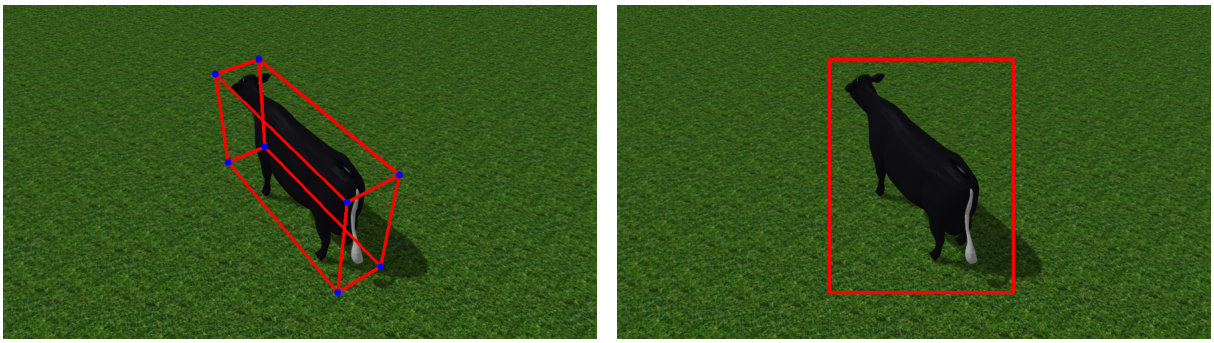
$$\begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}, \tag{5.18}$$

where $\mathbf{K}$ denotes the camera projection and intrinsic parameter matrix defined by:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.19}$$

where f_x, f_y denote focal lengths and c_x, c_y denotes the principal point in the image plane. Finally, image pixel coordinates (x,y) can then be obtained via similar triangles:

$$x = \frac{u}{w} \quad \text{and} \quad y = \frac{v}{w}. \tag{5.20}$$



(a) Projected 3D bounding box coordinates in the image plane.

(b) Resulting 2D object bounding box yielded from $min(\cdot), max(\cdot)$ on projected points.

Figure 5.3: ***Target Detection Instance Synthesis Example.*** *Example of instance synthesis used for providing training examples to the target detection network. Eight model-bounding world coordinates are expressed with respect to the camera reference frame and projected onto the image plane to form the final required 2D object bounding box provided as a single training instance.*

For each of the eight projected pixel coordinates (x_i, y_i), the final 2D image bounding box defined by $((x_a, y_a), (x_b, y_b))$ is generated via:

$$\begin{aligned} x_a &= \underset{i \in [0,8)}{\operatorname{argmin}} x_i, \\ y_a &= \underset{i \in [0,8)}{\operatorname{argmin}} y_i, \\ x_b &= \underset{i \in [0,8)}{\operatorname{argmax}} x_i, \\ y_b &= \underset{i \in [0,8)}{\operatorname{argmax}} y_i, \end{aligned} \tag{5.21}$$

where $i \in \mathbb{Z}$. This ground truth annotation is ultimately written to a Extensible Markup Language (XML) file – accompanied by the corresponding image – structured identically to the VOC format utilised in earlier chapters (see Section 3.3.2).

"Look At" Functionality

Whilst throughout training data synthesis, a single cow is placed at the origin and thus, camera angles C_θ, C_ψ are computed to centre the model in the image I, in reality these angles cannot be rawly computed – we do not know the positions of targets. As such, it is necessary to employ functionality to "look at" a target, centring it in the image. This is achieved by taking a chosen detection $RoI = ((x_a, y_a), (x_b, y_b))$ and computing it's centre point via:

$$\begin{aligned} b_x &= x_a + \frac{x_b - x_a}{2}, \\ b_y &= y_a + \frac{y_b - y_a}{2}. \end{aligned} \tag{5.22}$$

Similarly, the image centre point is also computed:

$$\begin{aligned} c_x &= \frac{I_w}{2}, \\ c_y &= \frac{I_h}{2}. \end{aligned} \tag{5.23}$$

Per-dimension pixel error between centres is then:

$$\begin{aligned} r_x &= b_x - c_x, \\ r_y &= b_y - c_y. \end{aligned} \tag{5.24}$$

Finally, error values are then used to find the relative angle between the image and RoI centre points, subsequently expressed as a clockwise north heading and issued as a camera yaw control value:

$$C_\psi = (atan2(r_x, r_y) + \pi) \bmod \pi. \tag{5.25}$$

The camera pitch angle C_θ is approximated via a simple proportional controller. A relative pitch angle displacement is found by:

$$\theta = K_p \cdot \sqrt{(c_x - b_x)^2 + (c_y - b_y)^2}, \tag{5.26}$$

where K_p denotes the constant proportional gain component empirically set to $K_p = 0.08$. For a full explanation of proportional controllers, refer back to Section 2.6.2. The final absolute camera pitch angle is fulfilled by taking the current camera pitch angle into consideration $C_\theta := C_\theta + \theta$.

5.4.2 Agent-Target Estimation (ATE)

For this process, the goal is to learn to approximate the function in equation 5.12:

$$\xi(I^i, C^i) \longrightarrow \vec{D}^i. \tag{5.27}$$

where I^i, $C^i = (C_\theta, C_\psi)$, $\vec{D}^i$ denote the target RoI image, camera gimbal parameters (pitch and yaw) and agent-target relative vector all at time-step i, respectively. That is, for a given camera gimbal configuration and a corresponding image (containing a target), what is the three dimensional agent-relative displacement vector between the agent and observed target in the agent's reference frame?

Due to the relative image simplicity in this process operating in simulation, it is achieved with a slight architectural modificaiton of simple CNN AlexNet [171], similarly to later Chapter 6. The change is made to permit the input of dual value tuple C^t as well as the image I^t into the DNN. Secondary input is achieved via a 2-unit input layer and single fully connected layer with 64 hidden units. The output is then concatenated with the tensor yielded from the second fully connected layer in AlexNet. A final, linearly-activated output fully connected layer with 3 outputs $\vec{D}^t = (x_D, y_D, z_D)$ concludes the network

architecture. Figure 5.4 illustrates the employed architecture. Back-propagation and optimisation is achieved via Mean Square Error (MSE) loss and Stochastic Gradient Descent (SGD) for the regression task.

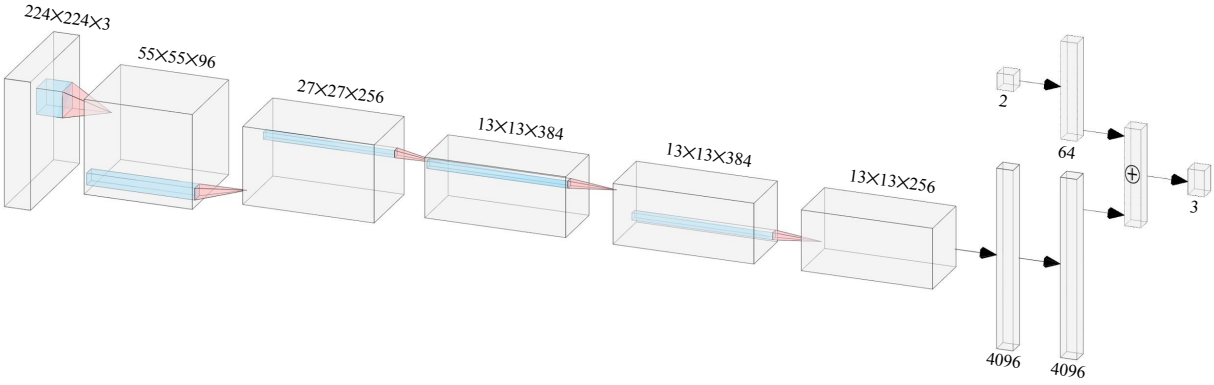


Figure 5.4: ***Agent-Target Estimation Network Architecture.*** *Modified dual-input CNN/DNN architecture based on AlexNet [171] for estimating the 3D vector between the agent and a visible target given an input* $224 \times 224 \times 3$ *image and corresponding camera pitch, yaw angles* C_θ, C_ψ*, respectively.*[3]

Estimation performance of the relative displacement vector between the agent and some target is assessed in the following paragraphs. Tests were conducted across $2,000$ newly generated instances. For a single instance and, similarly for training data synthesis, an agent position $\vec{G} = (G_x, G_y, G_z)$ is randomly generated in Local Active IDentification Network (LAIDN) bounds as according to equation 5.13. Additionally, a random target individual $r_i \in R$ with $i \in [0,9]$ is placed centred at the origin with random heading Cow_ψ. A model prediction is then compared against a generated ground truth label via MSE.

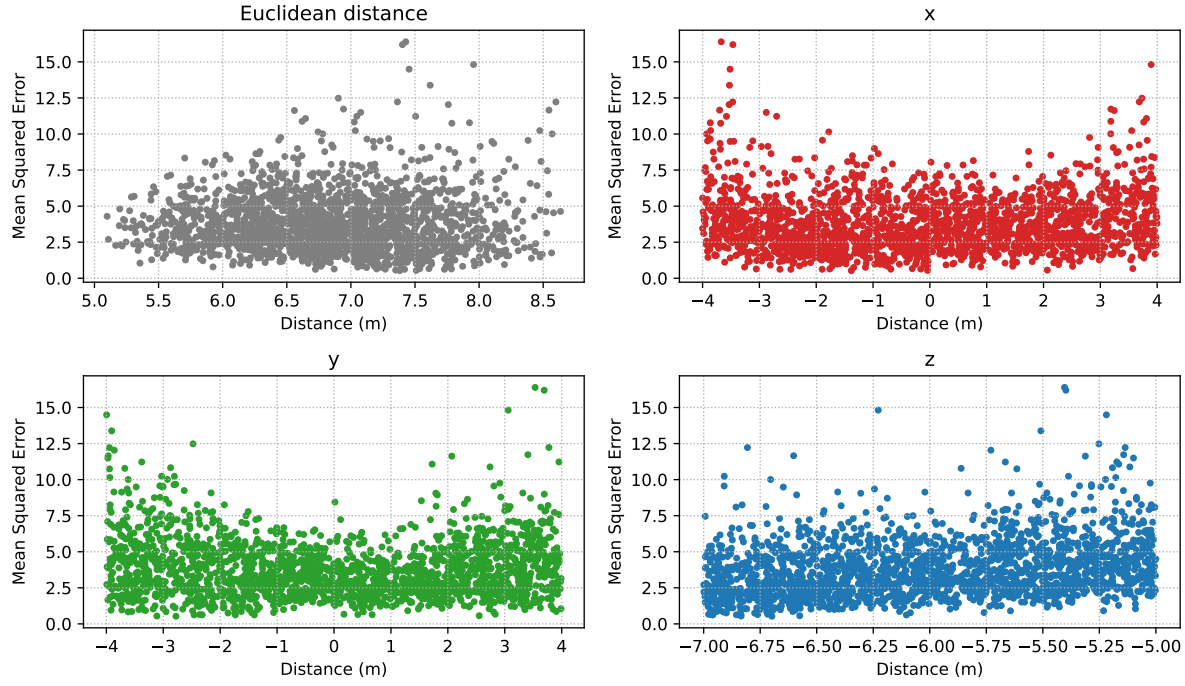


Figure 5.5: ***Distance Versus Mean-Squared Error.*** *Graphs illustrating the presence or absence of relationships between (top left): 3D Euclidean and (all other): dimension-wise agent-target distance and the corresponding MSE in the network estimated displacement vector* $\vec{D}$*.*

[3]Figure rendered using: `http://alexlenail.me/NN-SVG/`.

Figure 5.5 illustrates overall Euclidean distance and dimension-wise MSE versus the ground truth agent-target distance in the respective axes over the 2,000 testing instances. As can be seen in those graphs, the model demonstrates a slight tendency towards increased positional error in dimensions x, y over increasing distances owing to reducing object resolution. The general takeaway is that this component of the wider pipeline introduces relatively significant variable error. The result of which influences estimates of the target's position $\vec{A}$ and, will therefore affect the ability of the agent to realise newly sought viewpoints. This is demonstrably overcome in later experiments (see Section 5.5), however future work could integrate more precise monocular pose estimation methods [186, 13, 62].

5.4.3 Visibility Estimation (VE)

Within this process, the aim to is to establish which area, portion or segment of a target individual is currently being observed. The intuition being that once an agent is aware of the context of an observation, it can execute position-relative and individual-wise actions towards the observation of another, desired segment to solidify identity confidence. Put differently, if we theorise that we are currently observing the right side of r_i, and have some knowledge that this target has an identifying mark on its left side, we're able to suitably select actions that would lead to that left side (given knowledge of our own position).

Knowledge of the current form of target observation alone is powerful, but simultaneously limiting in that it fails to encode past observations. Complete output for this stage is consequently two-fold:

1. VE^0_{out}: of cardinality $|VE^0_{out}| = |VE_{classes}|$ subject to $\sum_{i=0}^{|VE_{classes}|-1} = 1$, this vector represents model confidence in the segment *currently being observed*.
2. VE^1_{out}: also of size $|VE^1_{out}| = |VE_{classes}|$ and with all values initialised to 0, this vector integrates current observation VE^0_{out} with past knowledge such that an understanding of *which segments have already been observed* is formed.

The secondary temporally-integrating output vector VE^1_{out} is computed in an additive manner;

$$VE^1_{out} := VE^1_{out} + VE^0_{out},$$
$$\text{where } \forall v \in VE^1_{out},\ v \in \mathbb{R},\ v := \begin{cases} 0 & \text{if } v < 0 \\ 1 & \text{if } v > 1. \end{cases} \tag{5.28}$$

A relatively simple model is adopted here for current target visibility estimation, as depicted in Figure 5.6. The problem is cast as a classification task with five possible classes to match the five visible faces

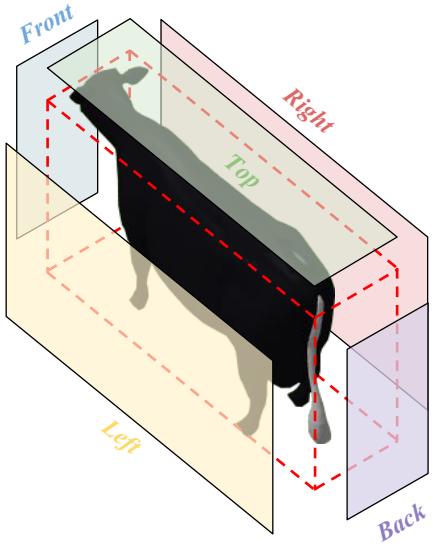


Figure 5.6: ***Cow Model Abstraction.*** *The employed parts-based representation/abstraction of cattle employed in this chapter. A three dimensional cuboid is fitted to spatially enclose the object. Each of five visible faces (e.g. 'top', 'front', 'back', 'left', 'right') are the possible classifications of which segment of the target the agent is current observing.*

of a cuboid placed on the ground:

$$VE^0_{out} \in VE_{classes} = \{\text{“top”}, \text{“front”}, \text{“right”}, \text{“back”}, \text{“left”}\}. \tag{5.29}$$

This decision, whilst adequately justifiable in the context of the task at hand (i.e. identifying arguably cuboid-shaped cows) generalises well to other classes of identification problems requiring fine-grained identification and retains model simplicity. Furthermore, it can easily be replaced with more complex solutions as required, given the modularity of the architectural design employed here.

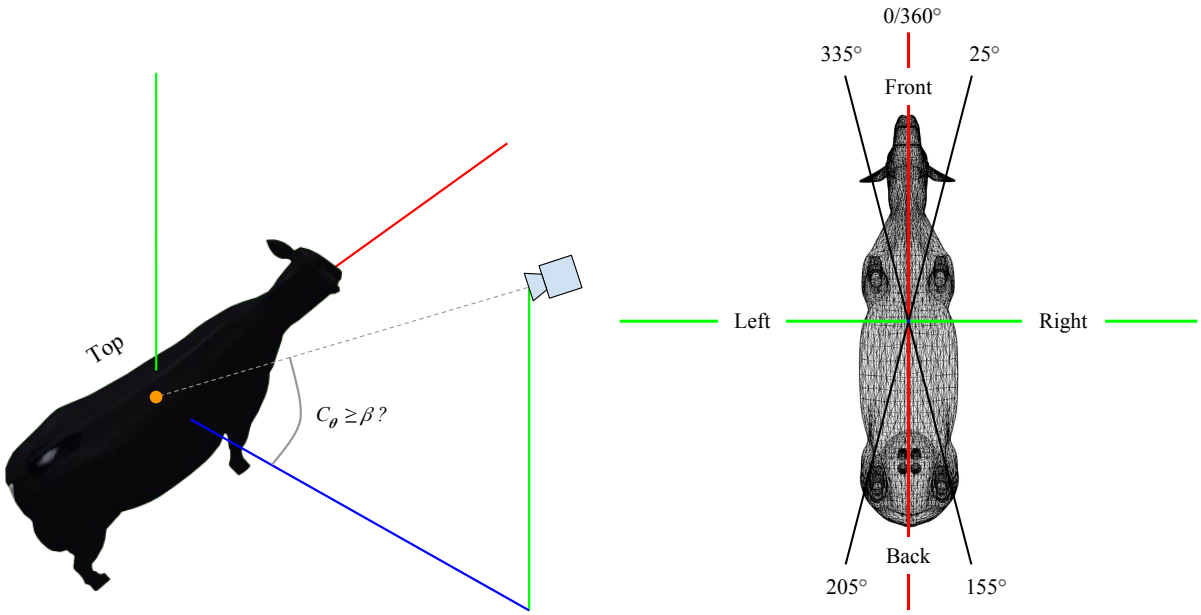


Figure 5.7: ***Ground Truth Target Segment Assignment.*** *The manner in which ground truth labels are assigned for a randomly generated training instance. (a): When $C_\theta \geq \beta, \beta = t_{pitch} = 60°$, the segment classification is ‘top’, otherwise (b): when the “look-at” pitch angle is too small, the corresponding visibility class is assigned via the heading intervals shown in this subfigure.*

Training data is generated by – very similarly to the other networks at this level – randomly generating an agent position within local active identification boundaries given by equation 5.13. The camera is then pointed at the object centre via equations 5.15 and 5.16. A random cow with random heading is placed at the origin. An image RoI containing the target is computed via the aforementioned target detection (TD) stage. Candidate RoIs are then binned into a class $c \in VE_{classes}$ via:

$$c = \begin{cases} \text{“top”}, & \text{if } C_\theta > \alpha \\ \text{“front”}, & \text{if } \psi < \frac{\beta}{2} \\ \text{“right”}, & \text{if } \psi < \frac{\pi-\beta}{2} \\ \text{“bottom”}, & \text{if } \psi < \frac{\pi-\beta}{2} + \beta \\ \text{“left”}, & \text{if } \psi < \pi - \frac{\beta}{2} \\ \text{“front”} & \text{otherwise,} \end{cases} \tag{5.30}$$

where heading angle ψ is generated via:

$$\psi = \begin{cases} |C_\psi|, & \text{if } C_\psi < 0 \\ (\frac{\pi}{2} - C_\psi) + \frac{\pi}{2} & \text{otherwise,} \end{cases} \tag{5.31}$$

to convert a computed yaw angle via atan2$(\cdot,\cdot)$ from the range $[-\frac{\pi}{2},\frac{\pi}{2}]$ to the clockwise heading/bearing range $[0,\pi]$ (see Figure 5.7b). Figure 5.7 illustrates the ground-truth label generation process. Threshold angle values $\alpha = t_{pitch} = 60°$, $\beta = t_{fr} = 50°$ for ground-truth labelling were empirically selected.

Network architecture is identical to that of the previous agent-target estimator (ATE, see Section 5.4.2, especially Figure 5.4) apart from the final layer. The final fully connected layer's activation function is simply changed to softmax for the classification task here, the number of output neurons increased to $|VE_{classes}|$ and a categorical cross-entropy loss function is used. $10,000$ training instances were synthesised for back-propagation with 10% retained as a validation set, achieving a best accuracy of 90.58%.

5.4.4 Identity Estimation (IE)

As established in Chapter 4, identification – even in a passive setting – demonstrably benefits from a multi-image process when operating on objects with fine-grained visual differences, such as Friesian cattle. Not only is this reflected in the motivation for this chapter, but this fact is capitalised on within this particular component tasked with identity estimation. As before, this refers to estimating the identity of the single individual contained within a set of image RoIs. As a result of the success demonstrated by the architectural choices made in earlier Chapter 4 (see Section 4.4), an almost identical architecture and data flow is implemented here as well. To recap, this consists of; *(1)*: GoogLeNet/Inception [302] – a very deep and wide CNN – extracting frame-wise feature maps[4] supplied to *(2)*: LSTM units[5] for temporal information integration. This form of spatio-temporal CNN/LSTM combination is referred to as a LRCN architecture, first proposed by Donahue et al. in 2015 [73].

LRCN Model Training Process

Actually training this LRCN-based model is non-obvious and relies upon a 2-stage process for both training data synthesis and model training. Fine-grained implementation details for the staged training approach are given as follows, whilst listing 2 summarises the approach algorithmically.

1. **CNN training data generation**: The task here is to synthesise/generate appropriate training data for GoogLeNet/Inception towards the goal of individual-wise feature extraction. To do so, the agent is placed at a randomly generated position within local active identification bounds (l,w,h_l,h_u) as according to equation 5.13. The camera gimbal is pointed at a point situated slightly above the origin (the centre of mass of the cow target model). A random cow model is placed centred about the origin with a random heading angle. A single training data instance then consists of a $224\times224\times3$ RGB input image yielded from the YOLO detector (see Section 5.4.1) and the corresponding $|R|$-D ground truth identity vector label. n of these single instances are synthesised to yield the final CNN training dataset.
2. **CNN model training**: The Inception CNN is then trained end-to-end for 50 epochs via SGD with momentum [250] using categorical cross-entropy loss and a fixed learning rate $e = 0.001$ on the n-strong dataset synthesised above. Of those instances, 10% are retained as a validation set.
3. **LSTM training data generation**: With the outer CNN trained for identity-wise feature extraction, training data can now be synthesised for LSTM training. This is achieved by again randomly placing the agent within local active identification spatial bounds. A random target with random heading is again placed at the origin. However now, the agent is randomly displaced $s_{fv}-1=4$

[4] Feature maps are a 1024-D sized vector computed by averaging the (x,y) axes for the $7\times7\times1024$ sized vector yielded from the last dropout layer following 2-dimensional average pooling (the "pool 5b" layer). Feature vectors are then concatenated into a single $s_{fv}\times1024$ temporally-consistent vector with $s_{fv}=5$ to match the number of abstracted object segments (e.g. top, front, back, left, right).

[5] The recurrent architecture used here consists of a single LSTM layer comprised of 256 units with dropout probability 0.2.

times within the defined boundaries with the cow model remaining static to match the number of defined object segments (i.e. top, front, back, left, right). At each position, the camera is pointed at the origin and the detected object RoI (via YOLO) is passed through the trained Inception CNN up until the final fully connected layer and averaged (as described above in Section 5.4.4) yielding a 1024-D feature vector for each image. These feature vectors are concatenated to form a single 5×1024 input training vector. As before, the corresponding identity vector of size $|R|$-D is included as the instance's ground truth label and m training instances are synthesised.

4. **LSTM model training**: Finally, the LSTM is trained on the dataset generated from the previous stage with the Adam optimiser [163] and categorical cross entropy loss. As before, the model is trained for 50 epochs and 10% of instances are retained as a validation set.

Algorithm 2 LRCN Training Algorithm. Algorithm for synthesising and training the employed LRCN via a staged approach.

for i **in** n **do**
  $img, label \leftarrow generateIDInstance()$
  $data_{CNN}[i] \leftarrow [img, label]$
end for
$CNN \leftarrow trainNetwork(data_{CNN})$
for i **in** m **do**
  $img, label \leftarrow generateIDInstance()$
  for j **in** s_{fv} **do**
    $features[j] \leftarrow CNN(img, until\_layer = "pool\_5b")$
    $img \leftarrow randomlyMoveAgent()$
  end for
  $data_{LRCN}[i] \leftarrow [features, label]$
end for
$LRCN \leftarrow trainNetwork(LRCN, data_{LRCN})$

5.4.5 Local Active Identification (*LAIDN*)

Network Architecture

The goal here is to suggest a new viewpoint to be realised by the agent that identifies the target in question as quickly as possible. This position is formulated based on identity information from past and current observations of the individual in combination with the context of the current observation; the current spatial relationship between the agent and the target. Consequently, inputs to the LAIDN consist of vector outputs from predecessor networks concatenated into a single 22-D vector with size $3+5+5+5+4=22$:

$$LAIDN_{in} = [ATE_{out} \oplus VE^{0}_{out} \oplus VE^{1}_{out} \oplus IE_{out} \oplus EA_{next}], \quad \text{with } |LAIDN_{in}| = 22. \tag{5.32}$$

Network output is simply a three dimensional position displacement vector in the agent's reference frame:

$$\vec{LAIDN}_{out} = (\Delta G_x, \Delta G_y, \Delta G_z). \tag{5.33}$$

Fulfilling this agent-relative position is trained to maximise the identification confidence in the current target. Since the input and output vectors are dimensionally simple, the corresponding network architecture can follow suit. As such, a network architecture consisting of an input layer followed by two fully connected layers with 64 neurons is used here, yielding a simple MLP-based ANN. The employed network architecture is illustrated in Figure 5.8.

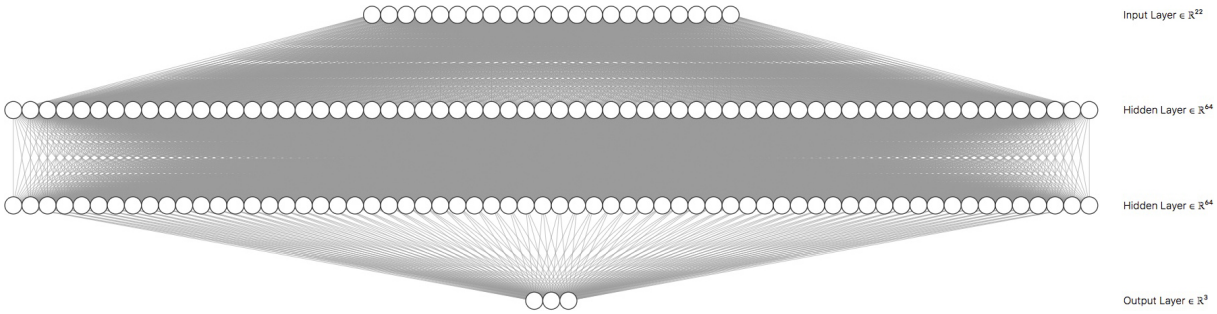


Figure 5.8: ***LAIDN Architecture.*** *ANN/MLP architecture for locally identifying a target individual. Input is a 22-D vector consisting of concatenated inputs from four predecessor networks whilst output is a 3D vector specifying a agent-relative goal position.*[6]

Agent Exploration Considerations

In this section, implementation details are given for the algorithmic integration of the subsequent Chapter 6 (outlining environment exploratory agency) within the complete, unified framework performing active identification proposed in this Chapter. Without having to refer to that chapter directly, the intention is to integrate the cost of fulfilling the next exploratory action into the evaluation of an active identification solution. Put differently, after identifying a particular target individual, the agent performs a new exploratory action towards discovering other individuals to be identified. Fulfilment of that new action should be considered when evaluating the efficiency of an active identification solution (an ordered sequence of viewpoints). Possible exploratory actions $a \in A$ performed by the agent are the four possible discrete navigation directions for a two-dimensional plane: forward, backward, left or right.

The following figure (5.9) illustrates the motivation for the inclusion of the subsequent exploratory action into identification configurations where a single image, single iteration is insufficient. Assuming that both paths – consisting of sequences of observations – successfully identify the individual in question (the LAIDN is confident in the correct identity over threshold value α), the right-hand path is clearly less costly in distance and therefore overall identification time. This is true in consideration of the exploratory action to be taken post identification, thus motivating the consideration of this factor in calculating the cost of an implemented identification path.

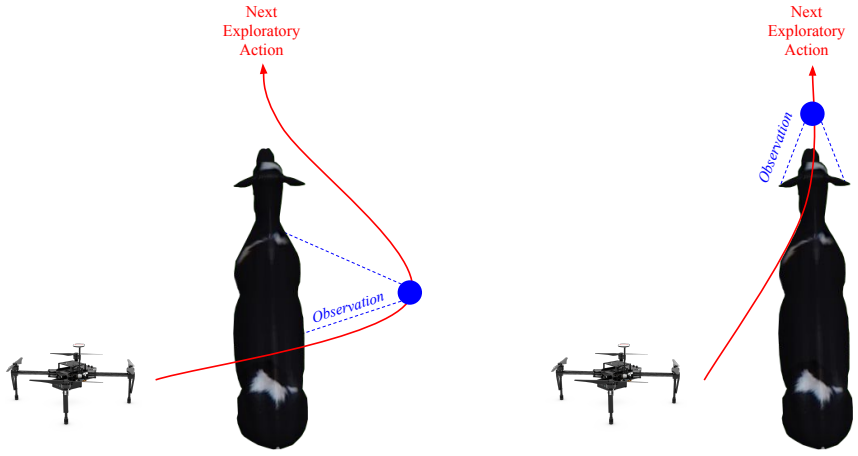


Figure 5.9: ***Flight Path Cost Considerations.*** *The exploratory action carried out subsequent to identification should be considered during path cost/optimality calculation. (Left): this path is more costly in distance (and therefore time) than the flight path shown in (right) due to the observation point not being spatially intersected by the path the UAV would take between its current and goal position dictated by the subsequent environment exploration action.*[7]

[6] Figure rendered using: `http://alexlenail.me/NN-SVG/`

Cost Function Design

In order to generate suitable training data for local active identification, an understanding of what constitutes a 'good' solution must be established. As is common in machine learning, a domain-specific cost function is designed here. In relation to the desired experimental outcome at hand, as mentioned previously, first and foremost, there is an obligation to identify an individual with confidence above a threshold value $\alpha \in (0,1), \alpha \in \mathbb{R}^+$. Secondarily, and the crucial consideration during training data synthesis, there is a requirement to achieve this identity confidence as quickly as possible. This desire is motivated by the nature of utilising a UAV robot; overall flight-time is limited by battery capacity. In addition, there may be many environment areas requiring exploration, certain individuals may require longer than most for identification, etc. – these factors therefore justify achieving satisfactory identification quality as quickly as possible.

The cost of a generated LAIDN solution is directly computed via the estimated time required to complete it. One solution instance s consists of a starting position, $n-2$ subsequent viewpoints and finally a single subsequent exploratory action to be enacted expressed as relative displacement vectors:

$$s = \{LAIDN^{0}_{start}, \vec{LAIDN}^{1}_{out}, \vec{LAIDN}^{2}_{out}, ..., \vec{LAIDN}^{n-1}_{out}, \vec{EA}^{n}_{next}\}. \tag{5.34}$$

The cost of such a solution is computed to be a combination of *(1)*: the time taken to fulfil each displacement vector (fly between the resolved positions) finishing with the next exploratory action and *(2)*: at each LAIDN stop, the amount of time required to perform respective computation. For *(1)*; time between consecutive LAIDN viewpoints $i, i+1$ is approximated via an assumption of constant average velocity using Euclidean distance:

$$t(s_i, s_{i+1}) = \frac{\sqrt{\sum_{k=1}^{3}(a_k - b_k)^2}}{v}, \tag{5.35}$$

given that $v = \frac{d}{t}$, where a, b denote positions resolved within the global reference frame by:

$$a = \begin{cases} s_0 & \text{for } i = 0 \\ s_0 + \sum^{i} \vec{s}_{i+1} & \text{for } 0 < i \leq n-1, \end{cases} \tag{5.36}$$

and similarly;

$$b = s_0 + \sum^{i+1} \vec{s}_{i+2}. \tag{5.37}$$

Velocity value $v = 0.5m/s$ was empirically selected to represent the average velocity of the UAV flight controller fulfilling some target position from some starting position. Similarly, *(2)* is assumed to be constant, and a value of $\beta = 0.5s$ was empirically selected for average visual on-board inference and processing time per iteration. The overall cost ϑ of a LAIDN solution s with respect to time is then:

$$\vartheta(s) = \sum_{j=0}^{n-1} t(s_j, s_{j+1}) + \beta(n-2), \tag{5.38}$$

where as shown before in equation 5.34, n denotes the number of LAIDN stops/viewpoints culminating in the fulfilment of a subsequent exploratory action to be taken.

[7] UAV image credit: DJI

Training Data Generation

Synthesising training data for this network – performing active identification on individuals – is considerably more involved than predecessor networks. Firstly, as usual, the agent is randomly placed within boundaries via equation 5.13. The camera is pointed at the object centre using equations 5.15, 5.16 and a random cow with random heading is placed at the origin. A target RoI is generated from the corresponding rendered image using the target detector (TD). The sub-image is subsequently given as input into the identity estimator (IE), if it successfully predicts the correct identity with confidence above the threshold α, this single image, single iteration was sufficient, thus a new configuration is randomly generated repeatedly until this is no longer the case. Otherwise therefore, this particular configuration requires active identification. In this case, LAIDN training instance generation commences.

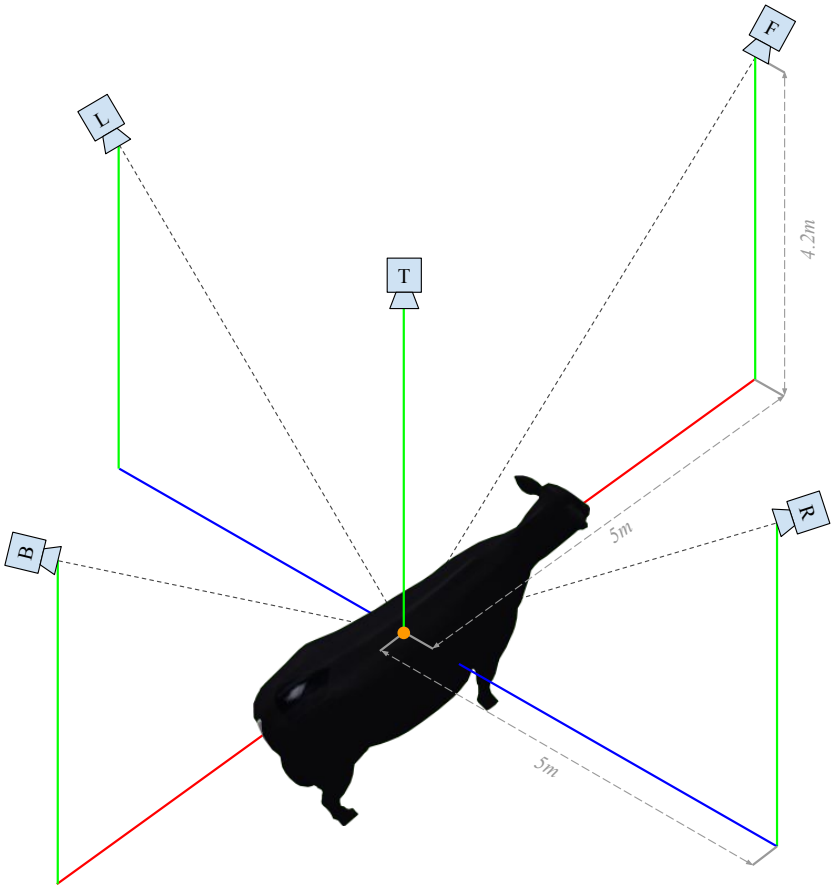


Figure 5.10: ***Optimal Segment Viewpoints.*** *Visualisation of target-relative optimal viewpoint definitions for each of 5 possible segments $v \in VE_{classes}$, where $VE_{classes} = \{\text{“top”}, \text{“front”}, \text{“right”}, \text{“back”}, \text{“left”}\}$. These position definitions are given as examples for LAIDN training example synthesis (see Section 5.4.5 for a full explanation of this process).*

Single instance synthesis is outlined as followed. Firstly, for a randomly generated agent position $\mathbf{A}_0 = (x_a, y_a, z_a)$ in LAIDN bounds, the corresponding image I and camera parameters C_θ, C_ψ are given to the visibility estimation network (see Section 5.4.3) to determine which cow segment is currently being observed:

$$v = VE(I, C_\theta, C_\psi) \tag{5.39}$$

where $v = VE^0_{out}$ is a one-hot vector encoding some element of $VE_{classes}$. This current segment observation vector v then forms the root of an acyclic, directed tree $T = (V, F)$. The tree is grown exhaustively for all permutations of remaining segment visitation orderings given the root segment v. This yields a tree with $|V| = 65$ (since $|VE_{classes}| = 5$), with every immediate parent-child vertex coupling having a

hamming distance of 1. As part of defining a new segment to be observed (excluding the root node), node attributes are accompanied by a 3-dimensional position that optimally views that segment (given camera parameters e.g. FoV, image resolution) – see Figure 5.10. An example of the grown tree is given below in Figure 5.11 with $|VE_{classes}| = 5$, as established in equation 5.30. Connecting edge attributes define the Euclidean distance between parent and child 3-dimensional waypoints later forming components of solution cost computation via equation 5.38. To conclude tree growth, the ending position:

$$\mathbf{A}_n = \mathbf{A}_0 + \vec{EA}_{next} \tag{5.40}$$

that fulfils the next exploratory action $\vec{EA}_{next}$ given the LAIDN starting position $\mathbf{A}_0$ is added as a new vertex to all sink nodes along with respective connecting edges (as depicted in Figure 5.11). This final position is fulfilled by all branches.

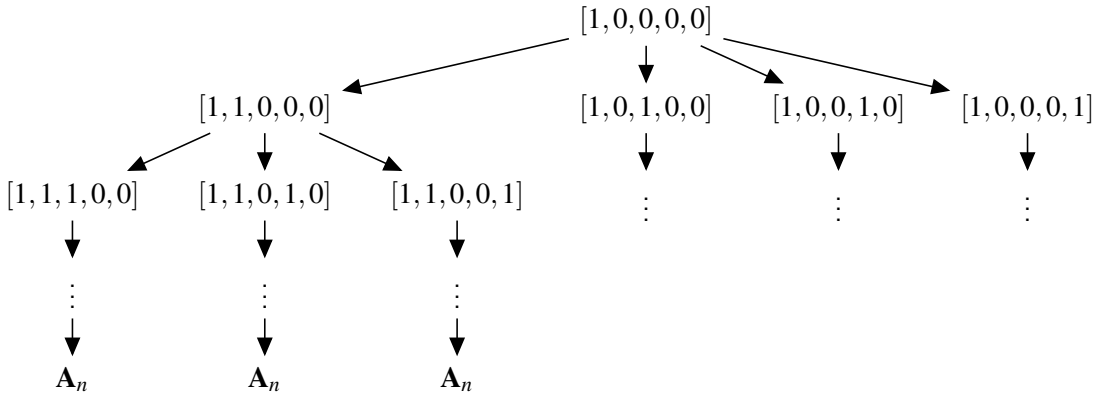


Figure 5.11: ***Segment Visibility Tree Growth.*** *Example of a training instance generation tree given an initial segment observation: 'top'. Nodes represent accumulative segment visitation down the connection depths of the tree in various orderings – the root note will have observed one segment only, whilst all sink nodes will have observed all segments and represent the position to be fulfilled according to the subsequent exploratory action $\vec{EA}_{next}$ given the LAIDN starting position $\mathbf{A}_0$. Every generated tree will always have depth $|VE_{classes}| + 1 = 6$.*

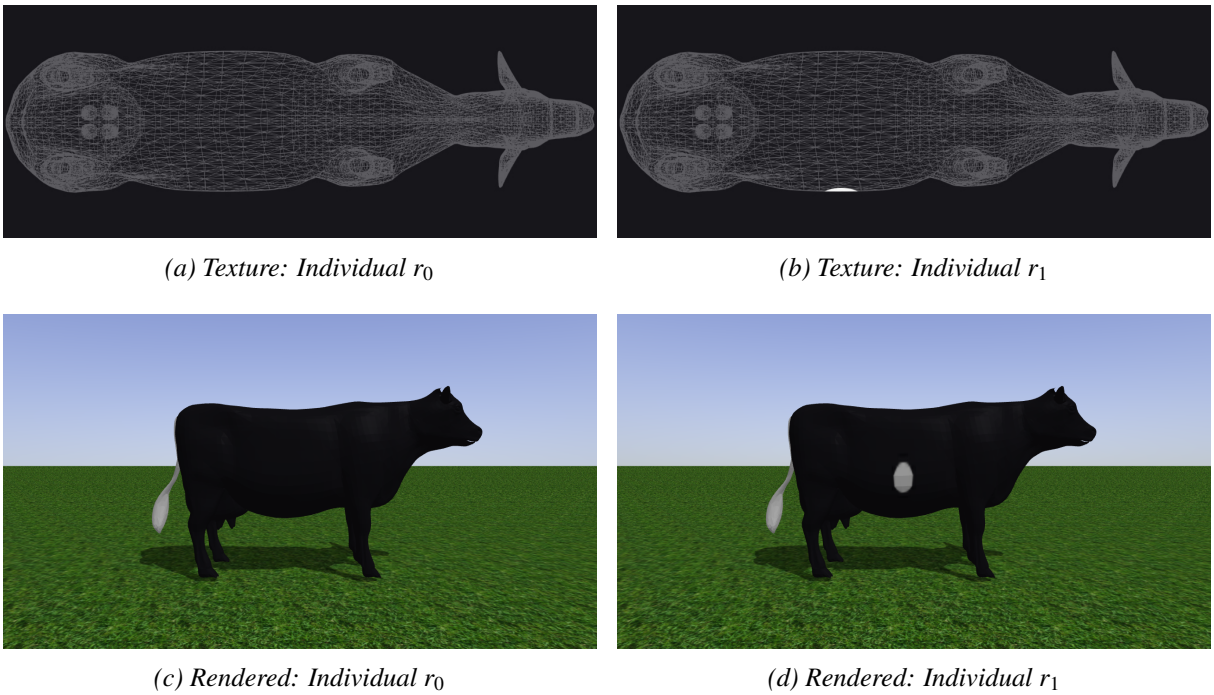
Following full tree growth, tree traversal occurs in order to determine the least-costly branch (defining an ordered set of spatial way-points/viewpoints) identifying the individual at hand to a satisfactory level ($\geq \alpha$), given the particular random initialisation (e.g. agent position, individual identity and heading orientation heading). This is achieved by determining all possible paths from the root source node to all sink nodes. These paths are then used to replay iterative identification in a passive setting; at each vertex (denoting a 3D waypoint/position that optimally views that segment) of a path, the image yielded as a result of pointing the camera at the model centre is added to a growing sequence of images. After each image addition, the last frame is copied to fill the remaining parts of the sequence such that a length $s_{fv} = 5$ is satisfied (see Section 5.4.4) and the sequence is given as input to the LRCN-based identity estimation network (IE). If the given image sequence satisfactorily identifies the individual, the cost of this solution (calculated via equation 5.38) is established as a baseline and traversal of that path terminates. For all subsequent path traversals (iterative identification replay), if the current cost exceeds the baseline, traversal ceases for that path. Conversely, if another path satisfactorily identifies the individual with lower cost, it is established as the new baseline solution. In this way, the least costly ordering of 3D viewpoints that identifies the individual at hand to a satisfactory level is extracted and provided as a singular training instance. The overall active identification solution is then separated into per-iteration individual LAIDN training instances, concluding the synthesis process for a single example.

5.5 Experiments and Findings

In this section, conducted experiments are described alongside corresponding analysis and discussion. With respect to this section and the previous describing model implementation, wherever relevant, 'random' numbers are generated using the Pseudo-Random Number Generator (P-RNG) Mersenne Twister algorithm [208]. This section is organised into experiments on a simple $|R| = 2$ population (Section 5.5.1) and second, the full herd identification experiment with ten contrived difficult identification cases is described in Section 5.5.2.

5.5.1 Baseline Experiment: 2-Strong Population Case

To commence experimentation and in order to verify the validity of the implemented active identification framework, a baseline experiment is conducted in this section. Whilst the experimental setup is relatively simple, it directly verifies whether the framework is able to extract and realise a good ordered sequence of low-cost observations that identify the target individuals to a satisfactory level. The setup consists of a population comprised of 2 individuals only (i.e. $|R| = 2$). Both cows are visually identical; having completely non-textured coats (black), apart from individual #1, who has one white mark placed on its right side (see Figure 5.12). Given this simple population $R = \{r_0, r_1\}$, in order to visually differentiate the individuals most efficiently, an agent should *always* intend to observe the right side of the current target. If the resulting observation contains a white spot, the individual is r_1, otherwise it is r_0. This experiment validates the question; with the active identification synthesis and training setup here, does automatic extraction and replication of this simple behaviour occur with low online cost?



(a) Texture: Individual r_0

(b) Texture: Individual r_1

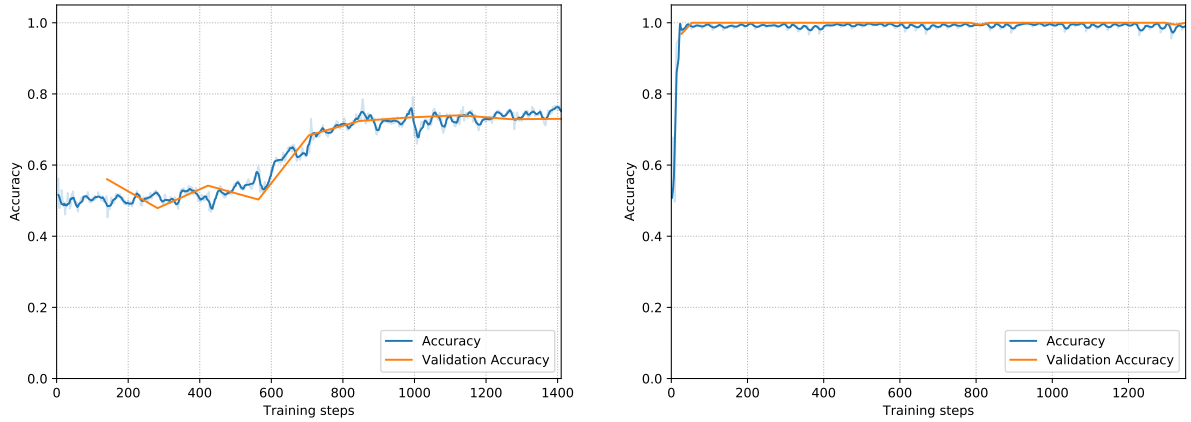
(c) Rendered: Individual r_0

(d) Rendered: Individual r_1

Figure 5.12: ***Individuals Forming the Two-Strong Population.*** *Depiction of the $(a), (b)$: individual textures with overlaid uv texture map and $(c), (d)$: corresponding renderings in the simulation environment (using Gazebo [164]) for the two individuals comprising the full population for the baseline 2-strong population experiment.*

In order to process this new population, some networks must be re-trained on new, appropriately synthesised data. This being said, existing model weights yielded from generic training was found to be sufficient for the: target detector (TD), agent-target estimator (ATE) and visibility estimator (VE) networks and thus, they were not re-trained. However, the identity estimator (IE) and consequently local

active identification (LAIDN) networks are not exempt from re-training necessity for trivial reasons. As such, the aforementioned synthesis and training procedures were followed (for *IE*, see Section 5.4.4, LAIDN Section 5.4.5). Training and validation accuracies versus training steps are illustrated in Figure 5.13 for *IE* and similarly, Figure 5.14 illustrates LAIDN MSE loss.



(a) Inception CNN *(b) LSTM*

Figure 5.13: ***2-Strong Population Identity Estimator Training Accuracy.*** *Training*[8] *and validation set accuracy versus training steps for the identity estimation network training on* (a)*: single image instances and* (b)*:* $s_{fv} = 5$*-strong random sequences of feature vectors extracted using the network trained for* (a)*.*

Interestingly, model accuracy in both the training and validation datasets for *IE* training on single frames (see Figure 5.13a) is observed to peak at the $\sim 1,200^{th}$ training step corresponding to $\sim 73\%$ accuracy. When considering the simplicity of the identification task on just two classes – as opposed to significantly higher accuracies achieved on larger population sizes in earlier Chapters 3, 4 – this value is peculiar. The cause is that randomly synthesised instances given as training to the identification estimation component themselves may be impossible to differentiate. Put differently, generated images may simply not contain the right side of individual necessary to differentiate the classes. Since generated camera positions in the y-axis are randomly distributed positively and negatively, approximately 50% of generated images will have the right side visible. Thus, the best the network can do in this scenario (occurring half of the time) is randomly guess the identity. Since there are two possible classes, it is correct half of the time yielding the achieved $\sim 73\%$ accuracy overall. This is overcome by the LRCN-based identified observing a maximum of five viewpoints, where the likelihood of one particular viewpoint containing the right-hand side is very high, as reflected in model accuracy in Figure 5.13b.

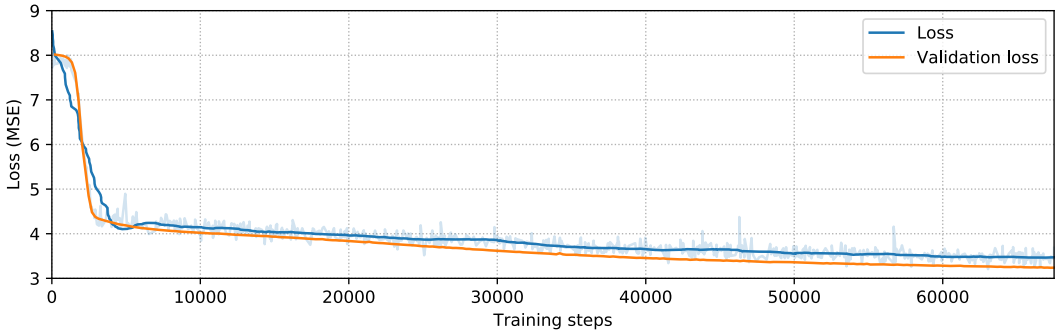


Figure 5.14: ***2-Strong Population LAIDN Training Loss.*** *Training and validation set MSE loss versus training steps for the local active identification network (LAIDN) training on* $9,000$ *examples, with an additional* $1,000$ *instances comprising the validation set.*

[8] For this graph and others in this chapter, the training signal was smoothed using the Savitzky-Golay filter [275].

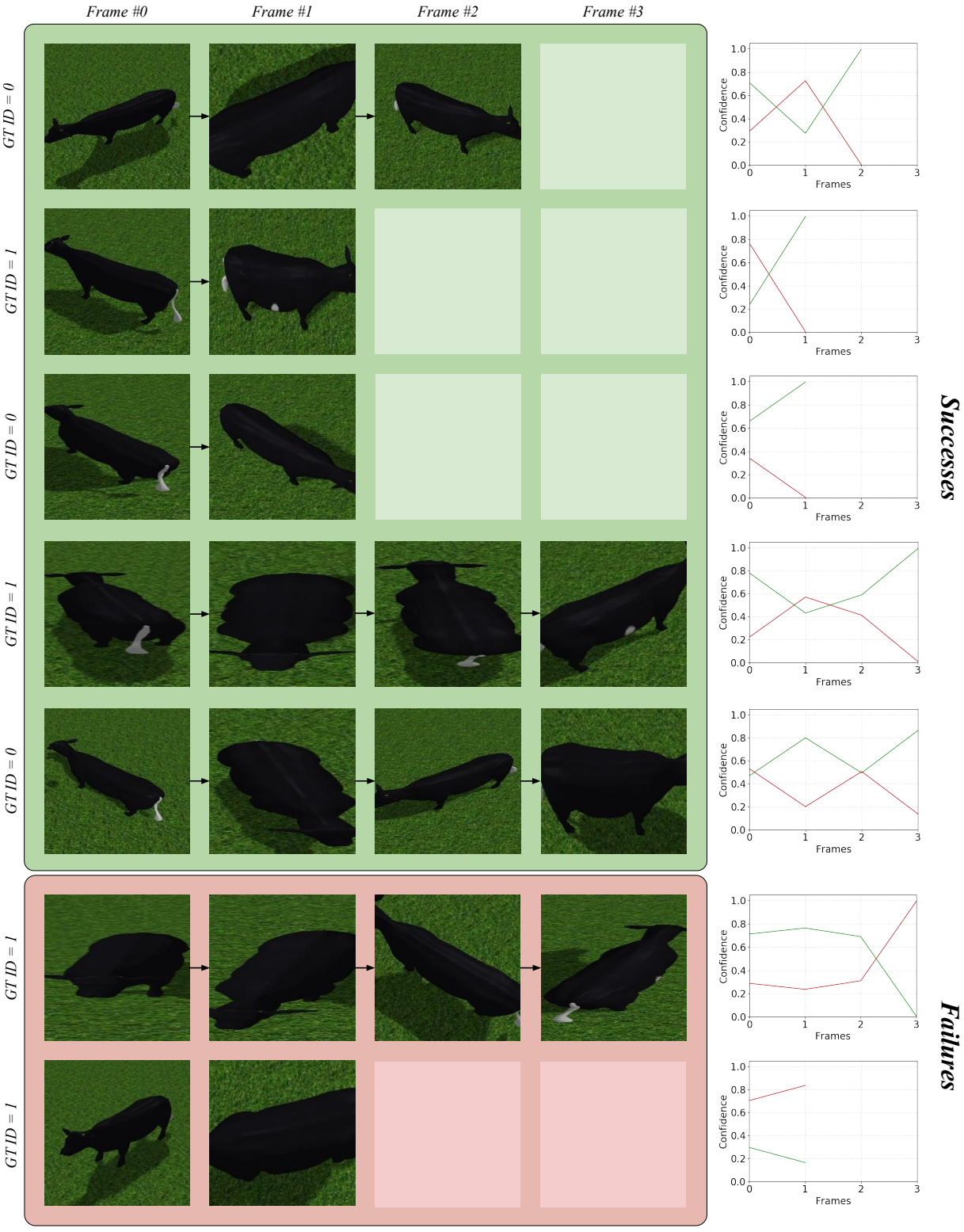
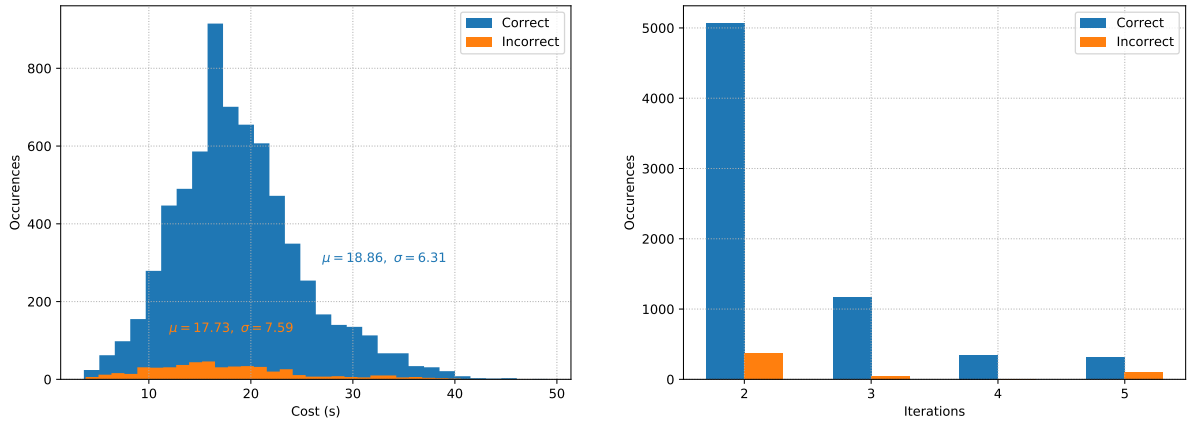


Figure 5.15: ***Local Active IDentification Network Baseline Experiment Successes and Failures.*** *Examples successes (top group) and failures (bottom group) for local active identification operating on the two-strong population case for randomly generated initial configurations. Ground truth identities are given on the left-hand side for observation sequences that terminate when the model is confident in some single identity to level* $\geq \alpha, \alpha = 0.8$. *Accordingly, graphs of model confidence in the (green): correct and (red): incorrect labels, respectively versus exposure to the number of frames is given for each instance in the right-most column. Since operation here is performed on just two individuals and softmax enforces output confidence vector summation to* 1*, these graphs are symmetric about confidence value* $k = 0.5$.

Row	Parameter description	Value	%
(a)	Total testing iterations	20,000	-
(b)	Overall correct	17,869	**89.35%**
(c)	Single iteration count	12,586	62.93%
(d)	Single iteration correct	10,983	87.26%
(e)	Single rhs visible	6,447	51.22%
(f)	LAIDN count	7,414	37.07%
(g)	LAIDN correct	6,886	**92.88%**
(h)	LAIDN rhs visible	3,620	48,83%

Table 5.1: ***Baseline Experiment Performance Statistics.*** *Table detailing performance statistics for the baseline experiment consisting of a population comprised of two near-identical individuals. Listed values denote (a): the total number of testing iterations and (b): the number of correct identity predictions, (c): of all the testing iterations, how many randomly generated configurations were deemed to be satisfactorily solved by the first observation (identity confidence $> \alpha$) and accordingly (d): how many were correct predictions, whilst row (e): denotes the proportion of those configurations in which the right hand side of the individual (containing the single spot or not) was actually visible. Finally, rows (f)-(h): similarly denote the frequency, performance and initial right side visibility when multiple iterations (and thus LAIDN) were deemed necessary to successfully identify the target individual, respectively.*



(a) Online LAIDN solution costs

(b) Number of viewpoints required

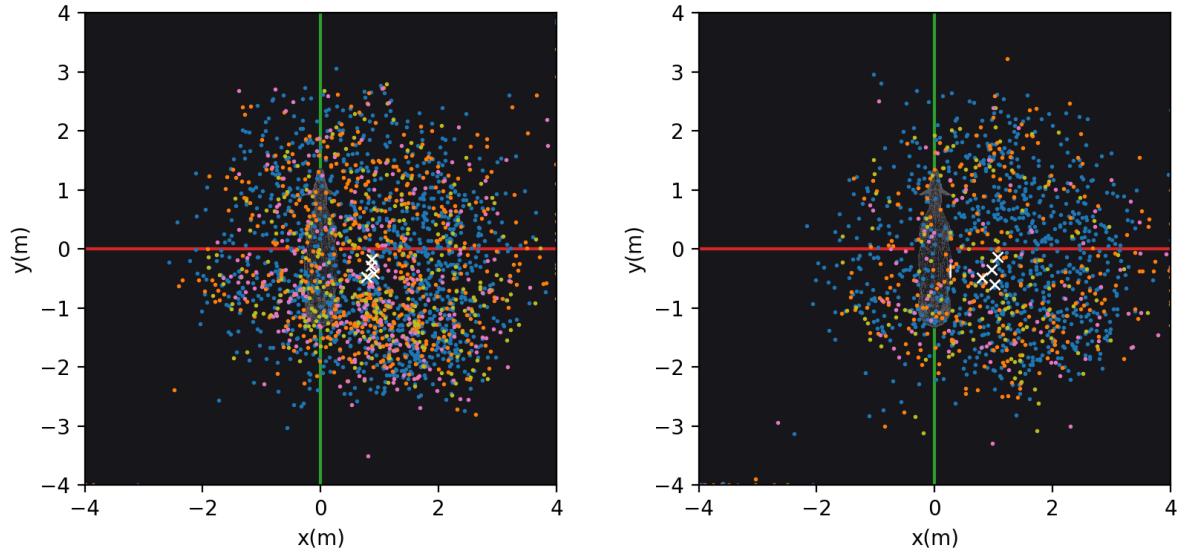
Figure 5.16: ***Baseline Experiment Histograms.*** *Histograms for the baseline experiment on an abstracted two-strong population case, where: firstly (a): of the $7,414/20,000 = 37.07\%$ test instances where the initial viewpoint was insufficient to identify the target individual and thus, active identification was required (refer to Table 5.1 for more statistics), what is the distribution of calculated solution cost (in seconds)? Second, (b): again for the $7,414$ testing instances requiring active identification, what is the distribution of new viewpoints (iterations) required to satisfactorily identify the subject with confidence $\geq \alpha$. Given the exceeding majority solve the testing instance in two iterations, this indicates success in the method reliably seeking to observe the target's right hand side, as confirmed qualitatively in Figure 5.17.*

In order to accredit an initial configuration with being solvable in a single iteration, the corresponding observation should have visibility of the right side of the individual in question to some extent. To deem whether this is the case, the randomly generated cow heading/yaw $Cow_\psi \in [0,360]^\circ$ and initial agent/UAV position $\vec{G} = (G_x, G_y, G_z)$ values are considered. First, the heading of the agent relative to the origin (where the target individual is placed) is determined:

$$h = atan2(y_a, x_a), \tag{5.41}$$

and is converted into a x-axis aligned heading value such that $h \in [0,360]^\circ$. If the resulting heading is within 90° either side of the individual's right side ($Cow_\psi + 90^\circ$), the configuration is deemed to be sufficient for successful single iteration, single observation evaluation. Table 5.1 illustrates achieved results of

this form by the active identification pipeline. As can be seen there, not only does the training algorithm reliably synthesise suitable example data, the proposed architecture is able to robustly perform efficient active identification in an online setting whereby the agent attempts to view the target's right hand side immediately (as can be observed qualitatively in Figures 5.15 and 5.17). This experiment then serves as a proof-of-concept in the proposed methodology operating successfully on a dummy case, intrinsically seguing into the following experiment involving a larger-sized population with difficult identification cases.



(a) $ID = 0$ *(b)* $ID = 1$

Figure 5.17: ***Active Identification Model Attention vs. Iterations.*** *Realised per-individual model viewpoints over multiple iterations for active identification testing instances highlighting iterative model attention. In these plots, the current target is placed at the origin with x, y axes indicated and $x, y \in [-4, 4]m$ (local identification boundaries). Iteration colour legend; (blue): iteration 2, (orange): iteration 3, (lime): iteration 4 and (pink): iteration 5. White cross markers illustrate cluster means for each iteration, indicating replication of the globally best strategy of seeking the target's right side. Note that the first iteration position (the initial agent position) is not shown here as it is just a uniform (pseudo-)random position in LAIDN boundaries and thus, reveals no information about the behaviour of active identification.*

5.5.2 Full Experiment: Simulation Results for Active Identification

In the full experiment described in this section, the population size is increased from the previous experiment consisting of two individuals to mimic a small-sized herd; $|R| = 10$. This 10-strong population is comprised of individually-unique *uv*-space textures that were manually created/painted to provide difficult identification cases. Identification difficulties arise from members of the population necessitating particular viewpoints in order to differentiate certain individuals. In some cases even, multiple viewpoints must be observed in any ordering for identification success to occur (examples of these identity pairs: $\{2, 3\}, \{4, 6\}$). The full synthetic population is illustrated in Figure 5.18. Justification of the choice to manually provide difficult cases requiring active identification in simulation resides within: first, in reality, an incredibly large population size would have to be registered in order to observe this behaviour (intra-population visual similarities increase with its cardinality). Second, operating in simulation allows extensive results to be concluded in a non operationally-costly setting such that, these results can serve as a proof-of-concept.

The experiment begins with re-training relevant models towards the new population. As for the two-strong population, components: TD, ATE and VE – solving target detection and localisation, agent-target 3D transform estimation and target segment visibility estimation, respectively – do not need to be re-trained here. The identity estimation network (IE) is re-trained on synthesised imagery utilising

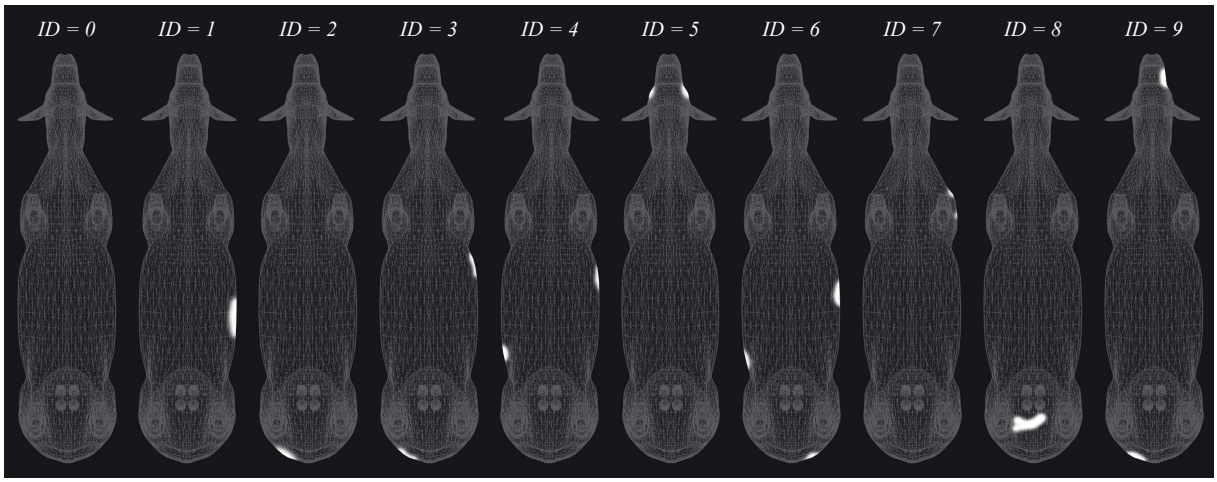
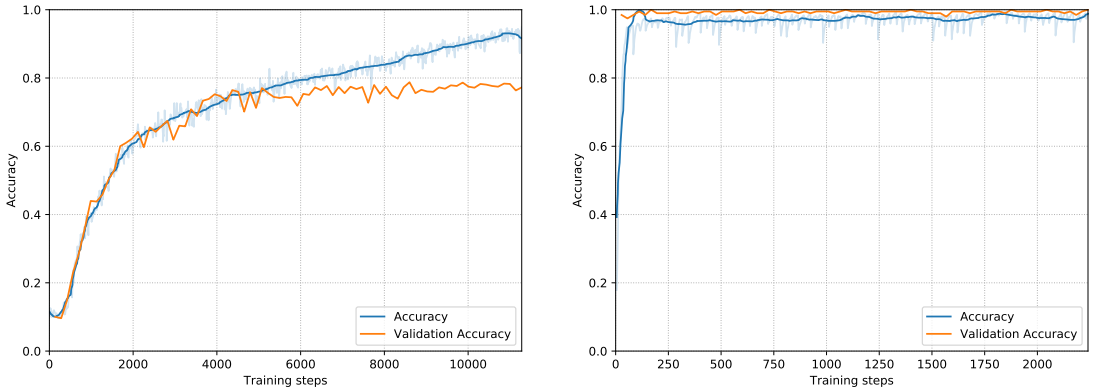


Figure 5.18: $|R| = 10$ ***Active Identification Population.*** *Depiction of the* $|R| = 10$ *textures and corresponding identities for the task of verifying the active identification pipeline proposed in this chapter. Each texture was manually created to yield difficult identification cases that require an active approach for differentiating identities.* $ID = 0$ *presents the only case with absolutely no markings. Importantly, note that this Figure depicts perfectly orthographic views for illustrative purposes (all features are visible) whereas within the simulator, the perspective camera model negates this possibility.*

the algorithm/process given in Section 5.4.4 and Algorithm 2. Model training accuracy versus epochs for this component is given in Figure 5.19, whilst results and accuracies of model training on this data is shown in Table 5.2. Note that the maximum image sequence length given as input into the LRCN is set to $s_{fv} = |VE_{classes}| = 5$, to match the number of possible object viewpoints (e.g. top, front, right, back, left). As this parameter selection matches the cardinality of the discrete viewpoint set $VE_{classes}$, five differing viewpoints should always be sufficient to identify the target subject (providing they are realised well). Next, training data for the network performing active identification (LAIDN) was synthesised and used to train the MLP architecture, again using the process described in details on the original implementation (see Section 5.4.5).

Row	Model	#Instances	Train:test ratio	Validation Accuracy (%)	Figure
(a)	*IE*	10,000	9 : 1	78.66	5.19a
(b)	LRCN	2,000	9 : 1	100	5.19b
(c)	LAIDN	20,000	9 : 1	68.43	5.20

Table 5.2: ***Model Re-training Results/Statistics.*** *Results and statistics for the model training process towards* $(a), (b)$*: identity estimation and* (c)*: active identification. These are the only models that require re-training when operating on a new population in simulation.*



(a) Inception CNN *(b) LSTM/LRCN*

Figure 5.19: ***Identity Estimation Accuracy vs. Training Steps.*** *Identity estimation model training*[9] *and validation set accuracies for* (a)*: the single frame evaluation CNN and* (b)*: the LRCN-based identification network operating on generated image sequences versus training steps over the given* $9,000/1,000$ *train/test instances.*

To validate the performance of the full active identification pipeline, 100,000 testing instances are assessed here. For each instance, an initial random agent position is generated local to the origin as according to equation 5.13, at which a randomly selected target identity with random heading/orientation is placed. Note also that the subsequent exploratory action to be fulfilled post-identification is also randomly selected from the set of possible actions (e.g. $A = \{f, b, l, r\}$, see subsequent Chapter 6 for further description of the exploratory framework). What follows are the results over all testing instances alongside corresponding analysis and discussion.

Row	Parameter description	Value	%
(a)	Total testing iterations	100,000	-
(b)	Overall correct	78,955	**78.96%**
(c)	Single iteration count	88,095	88.10%
(d)	Single iteration correct	71,771	81.47%
(e)	LAIDN count	11,905	11.91%
(f)	LAIDN correct	7,184	**60.34%**

Table 5.3: ***Online Performance Statistics.*** *Model performance statistics for online testing of* (a)*:* 100,000 *newly generated LAIDN scenarios yielding* (b)*: complete identity recovery correctness. Row* (c)*: gives the count and percentage of testing instances where the model was confident in some identity* $\geq \alpha$ *after seeing a single frame and the corresponding* (d)*: correctness of that prediction. When this is not the case,* (e)*: LAIDN is entered resulting in* (f)*: active identification correctness score on the synthetic and difficult* $|R| = 10$ *population.*

To begin, online performance results are given in Table 5.3 operating over the entire test set. In the vast majority of cases, nearly 90%, the random initial agent position satisfactorily identifies the target in a *single* iteration; $\sim 80\%$ of which are actually predicted correctly. This success is attributable to generated viewpoints containing large amounts of identifying features, despite random initialisation as a result of; (a): the implemented texture project method and (b): the population itself. For (a): and put differently, as can be seen in the full set of textures (see Figure 5.18), given a perfectly orthographic projection, a top-down view is the canonical viewpoint across the population. Whilst this perfection is not observed in reality due to feature resolution and camera perspective from the pinhole camera model, random initial viewpoints consisting of top-down imagery, particularly when centred above the target, still reveal salient identification features. In consideration of (b); the spatial distribution of features for the two-strong population meant that certain viewpoints (e.g. left side) gain no information whatsoever. This is no longer the case here, certain viewpoints now immediately eliminate candidate identities meaning the network can make a more informed estimate or guess. If this estimate is sufficiently confident, the evaluation of this single iteration is deemed satisfactory. As an example for the population here, were an agent to observe the left side of a random subject with no discernible markings, this immediately implies that the subject must be either one of $ID = \{0, 1, 7\}$.

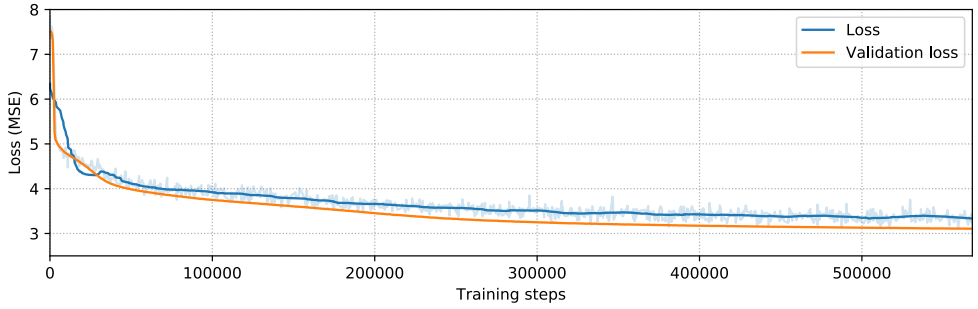


Figure 5.20: ***LAIDN Training Loss.*** *Graph illustrating LAIDN model training and validation MSE loss for* 1,000 *epochs on* 20,000 *training instances (*10% *of which are retained as a validation set).*

[9] Noisy training accuracy has been smoothed in both graphs and others in this chapter using the Savitzky-Golay filter [275] for visualisation purposes.

Across the 100,000-strong testing set, LAIDN was enacted 12% of the time, where, 60% of those instances were correctly predicted. Figure 5.21 illustrates how particular identities affect identification recovery performance. Notably, particular identities can be seen to cause high true and false positive rates from the identification difficulty levels owing from varying spatial distribution of features. Most interestingly is $ID = 0$ with no visual markings whatsoever, from which all other identities are founded upon texturally. This identity is observed to yield highest false positive rates across single and multi-frame evaluation. This is since many other identities are identical to $ID = 0$ bar one particular feature and thus, one particular observation/viewpoint. Instances where the agent observes multiple viewpoints containing no markings mean that model confidence in $ID = 0$ will be high, perhaps even above the threshold $\alpha = 0.8$. Future work will look towards varying this threshold and its effect on true and false positive rates.

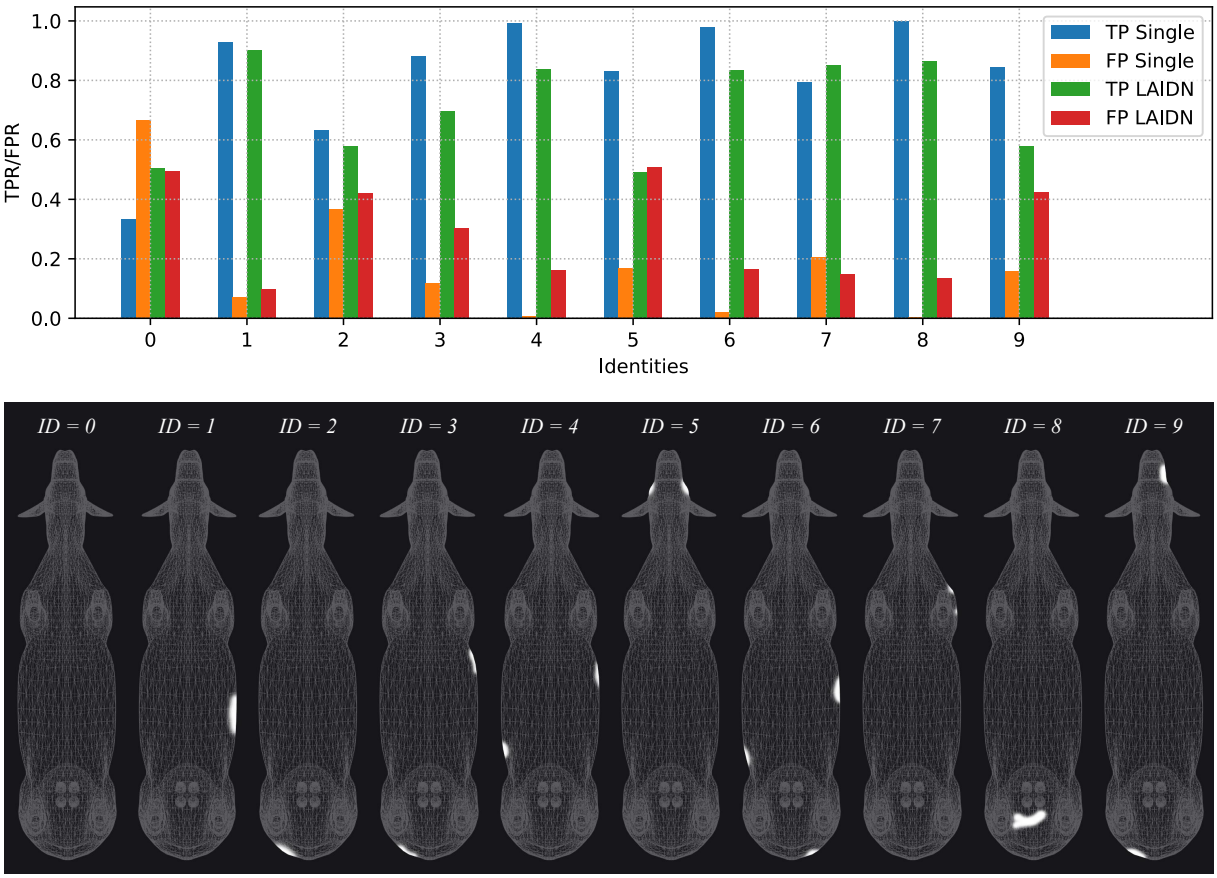
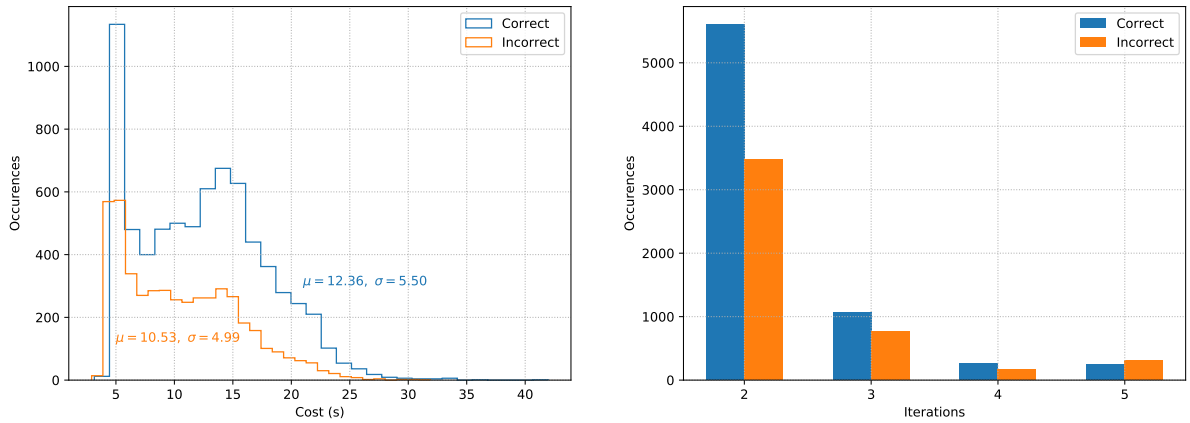


Figure 5.21: ***Identity-Wise Success/Failure Rates.*** *True and false positive rates for each identity for (blue, orange): single iteration evaluation and (green, red): instances where multiple iterations were required (hence using the LAIDN). The original population (bottom) are included here again for comparative purposes.*

Figure 5.22 depicts the distribution of online LAIDN solution costs as well as the distribution of correctness versus iterations. In those graphs, it is visible that incorrect LAIDN solutions have lower mean cost; calculated to approximate the time required by the agent to identify the target individual. This equates to LAIDN solutions finishing in fewer iterations and/or realising lower cost viewpoints that incorrectly satisfy confidence threshold α prematurely. In Figure 5.23, the spatial distribution of two-dimensional viewpoints realised by LAIDN iterations is illustrated alongside the per-individual likelihood for requiring certain numbers of active iterations to satisfactorily identify the target. Visible there are cases where the random initial single frame is exceedingly sufficient to identify the individual in question (e.g. $ID = \{4, 6, 8\}$), especially $ID = 8$, where any view of the top of the animal reveals its identity. This crucial feature is often visible for the randomly generated agent positions and thus, LAIDN is rarely entered for these individuals. Whilst cases that require multiple viewpoints to confirm identity have

high concentrations of realised agent positions/viewpoints across iterations (e.g. $ID = \{0, 2, 7\}$). As expected, the visible spatial distributions across iterations are far more chaotic and un-organised than the two-strong population case – merely viewing the target's right side is no longer the globally-best strategy across the population. Instead here, the LAIDN proposes agent displacement based on past and current estimates of target identity, and since there are many individuals and corresponding possible viewpoint sequences, suggested new positions demonstrate variability. This spatial variability per-iteration is then compounded by the random nature of the initial agent position along with variable per-dimension error from the network estimating agent-target displacement (see Section 5.4.2).



(a) Online LAIDN solution costs *(b) Number of required viewpoints*

Figure 5.22: ***Active Identification on*** $|R| = 10$ ***Population Experiment Histograms.*** *Histograms for* (a)*: online LAIDN solution costs when prediction was correct and incorrect over the* $11,905$ *testing instances where it was enacted and* (b)*: the distribution of iterations/frames/viewpoints required to satisfactorily identify the current target.*

To finish, Figures 5.24 and 5.25 then go on to illustrate a number of hand-picked LAIDN examples ultimately yielding identification estimation success or failure, respectively. Visible are examples where particular viewpoints yield erroneous identity estimates that are overcome (since confidence threshold $\alpha = 0.8$ is not satisfied) in later observations and vice versa. As can be seen in many of the examples, visibility of a particular individual's crucial identification feature is often very marginal. This is since the realised viewpoint reveals the presence of the feature with minimal cost; highlighting the success of the LAIDN training data synthesis algorithm in minimising generated solutions lengths and the identity estimation model's ability to correctly differentiate individuals with partial feature observability (often at low spatial resolutions). Note that this will also be a contributing factor to the seemingly arbitrary nature of yielded agent position distributions (as shown in Figure 5.23). In their entirety, the complete results obtained in this experiment – achieving 79% accuracy across $100,000$ test instances – indicate the success of the full identification recovery pipeline in performing robustly, even across a difficult set of synthetic individuals.

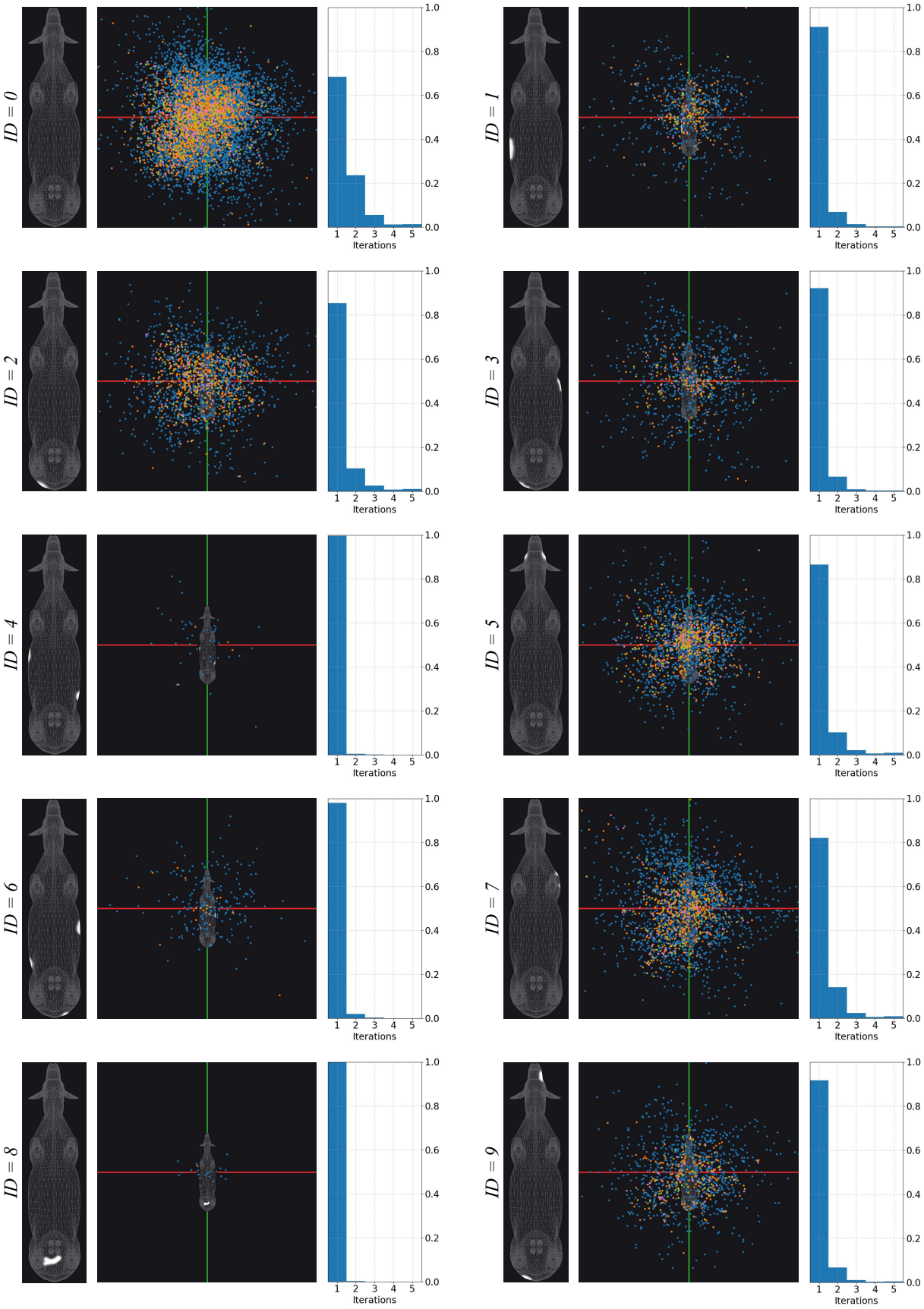


Figure 5.23: ***Identity-Wise Iteration Distribution.*** *Illustrations of per-identity iteration distributions. For each identity, (left): uv texture, (middle): spatial distribution of LAIDN iterations (> 1) and (right): normalised histogram for iterations required to satisfy identity confidence threshold $\alpha = 0.8$. For spatial distributions, (blue): iteration 2, (orange): iteration 3, (lime): iteration 4 and (pink): iteration 5. Note that the first iterations denoting the random initial agent position are not plotted here.*

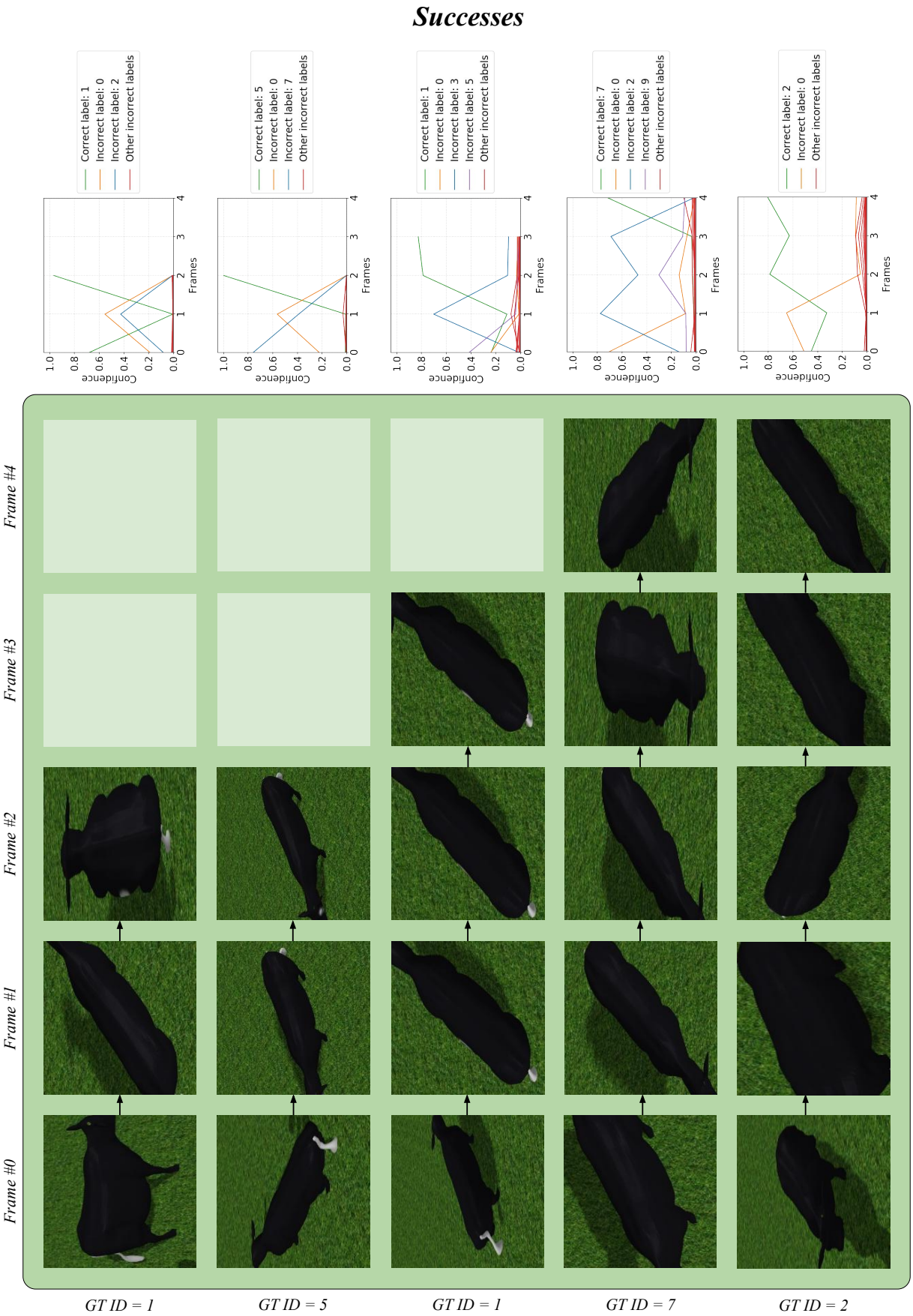


Figure 5.24: ***Example LAIDN Successes.*** *Hand-picked interesting examples of LAIDN successes involving the model actively seeking new viewpoints to satisfactorily identify a target. In each example (rows), the ground truth identity is given (refer back to Figure 5.18 for comparison) followed by constituent frames and the corresponding graph of identity estimation model confidence versus iterations (frames).*

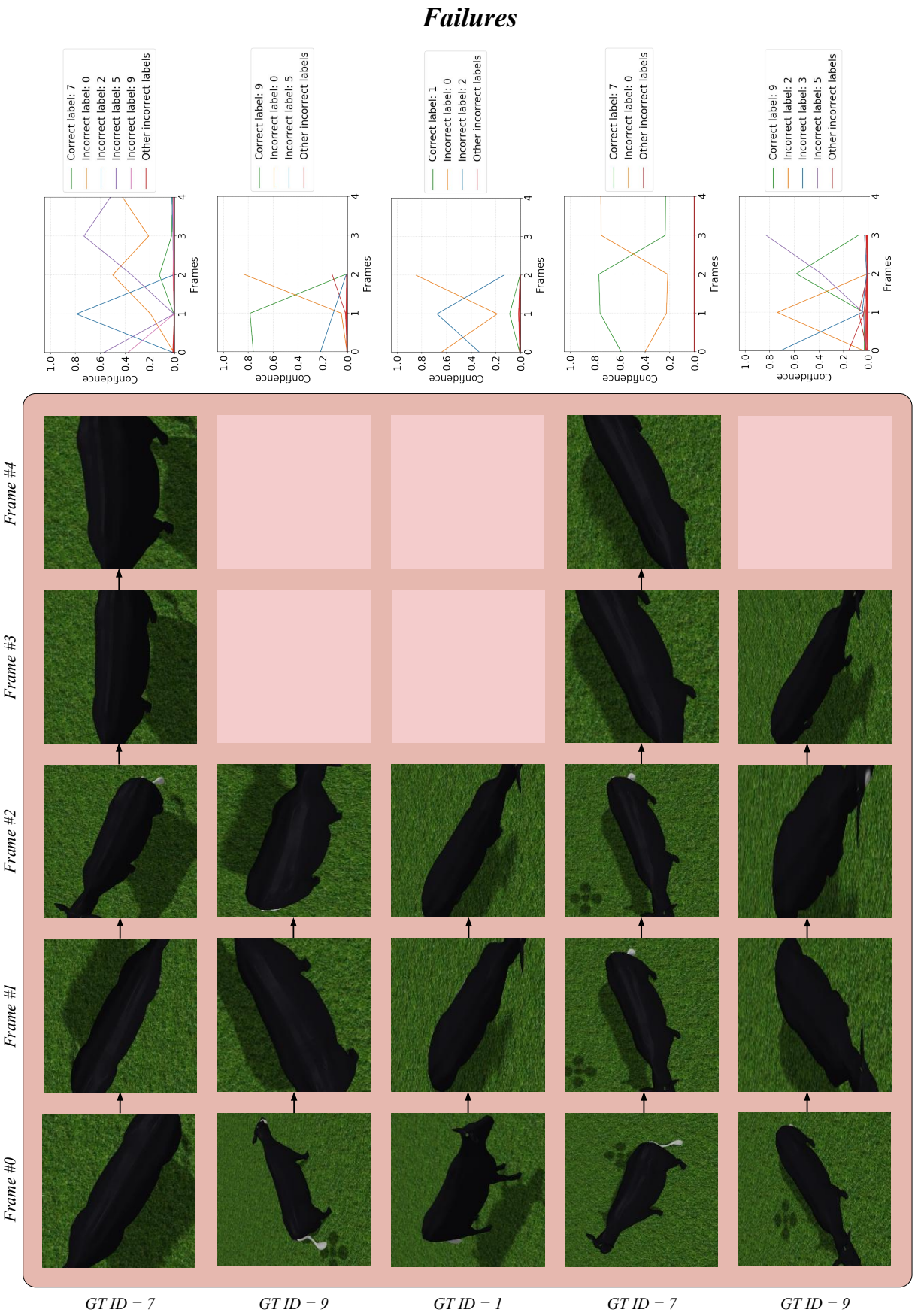


Figure 5.25: ***Example LAIDN Failures.*** *Identically to Figure 5.24, this figure presents a selection of observed failure cases across varying numbers of frames along with the identity estimation model's confidence versus exposure to subsequent frames.*

5.6 Chapter Conclusion

This chapter gives details for an active identification pipeline operating on realistic and static three dimensional cattle models in simulation. The proposed model is comprised of several foundational networks that approximate the agent's surroundings and the context of its current identification scenario. Approximations allow an attentive child model to suggest new viewpoints that optimise identification success whilst minimising solution cost. Across simulated cases, performance of this operational paradigm is demonstrably strong across a baseline experiment with two almost identical individuals. This operational success generalises to a small population size of synthetic individuals designed to create difficult identification cases. These experiments provide a proof-on-concept that the proposed framework can robustly extract and replicate a sequence of individual-specific and context-specific viewpoints that identify the target as quickly as possible.

In aligning this chapter within the wider context of this thesis as a whole, the task of identifying a single particular individual is now solved in multiple ways, concluding the first part of this thesis. This has been progressively demonstrated across *(1)*: single frame evaluation, *(2)*: passive frame sequence evaluation, and in this chapter, *(3)*: frame sequence evaluation with active, per-situation viewpoint selection. When an active identification paradigm is not completely necessary (e.g. a single dorsal image of the individual in question is sufficient to identify it with respect to some larger population set), earlier chapters provide robust architectures to solve these simpler cases. These solutions assume the agent is local to the individual in question; the target is visible and thus, can be detected – performing *local individual identification*. However, the intention of this thesis is to identify the group collectively, therefore, the agent needs to perform a global action to discover new targets to identify. It is this task – actively navigating an uncharted environment towards efficiently discovering unvisited individuals – that the subsequent chapter will explore.

Chapter 6

Simulated Inter-Individual Navigation

6.1 Chapter Overview

This chapter discusses deep learning for solving static and dynamic search and recovery tasks – such as the retrieval of all instances of actively moving targets (cows) – based on partial-view, UAV-like sensing. In particular, abstracted tactic and strategic exploratory agency is demonstrably implemented effectively via a single deep network that optimises in unity: the mapping of sensory inputs *and* positional history towards navigational actions. A dual-stream classification paradigm is proposed that integrates a first CNN for sensory processing, with a second for interpreting an evolving long-term map memory. In order to learn effective search behaviours, given agent location and agent-centric sensory inputs, this design is trained against optimal navigational decision samples for different multi-target distribution classes. Recovery performance is quantified across an extensive range of scenarios; including probabilistic placements and dynamics, as well as fully (pseudo)random target walks and herd-inspired behaviours. Detailed results comparisons show that the design can outperform naïve, independent stream and off-the-shelf DRQN solutions. Altogether, the proposed dual-stream architecture can provide a unified, rationally motivated and effective architecture for solving online search tasks in dynamic, multi-target environments.

6.2 Introduction

In this chapter, a map-based, unified deep learning framework (see Figure 6.1) is proposed; applicable to recovery tasks in structured environments where an agent with local sensing and spatio-temporal long-term memory is tasked with visiting, within a confined space as quickly as possible, *each* of a known-sized set of static or dynamically moving targets. In the special case where agents return and target locations are fully known over time and space, the task can be mapped to the static or dynamic Travelling Salesman Problem (TSP) [130], respectively. Practical solutions for this problem class have in the past been computed using Dynamic Programming [129], Ant Colony System optimisation [74, 88, 53] and Evolutionary Computing [95, 120, 141, 311] and more [285, 103, 152]. Within this chapter, a more realistic scenario is considered: where target locations are initially unknown, consequently requiring exploratory agency. In the wider context of this thesis, this equates to: given an unexplored field containing cattle individuals we'd like to find and identify, what search strategies should be employed, with efficiency in mind?

The described recovery task may classically be interpreted as a Partially-Observable Markov Decision Process (POMDP), described in various surveys on visually-motivated robotic navigation [34, 68] prior to deep learning. However in this chapter, the task is proposed to be cast into a framework for optimising deep recurrent classification. That is, mapping positional history and current sensory inputs to new navigational actions via a single DNN. Similar to Zhang et al. [345] in their recent work on reinforcement learning for exploration, explicit long-term memory is integrated into the design, experimenting with both storage of spatial as well as spatio-temporal information – as is depicted in Figure 6.1.

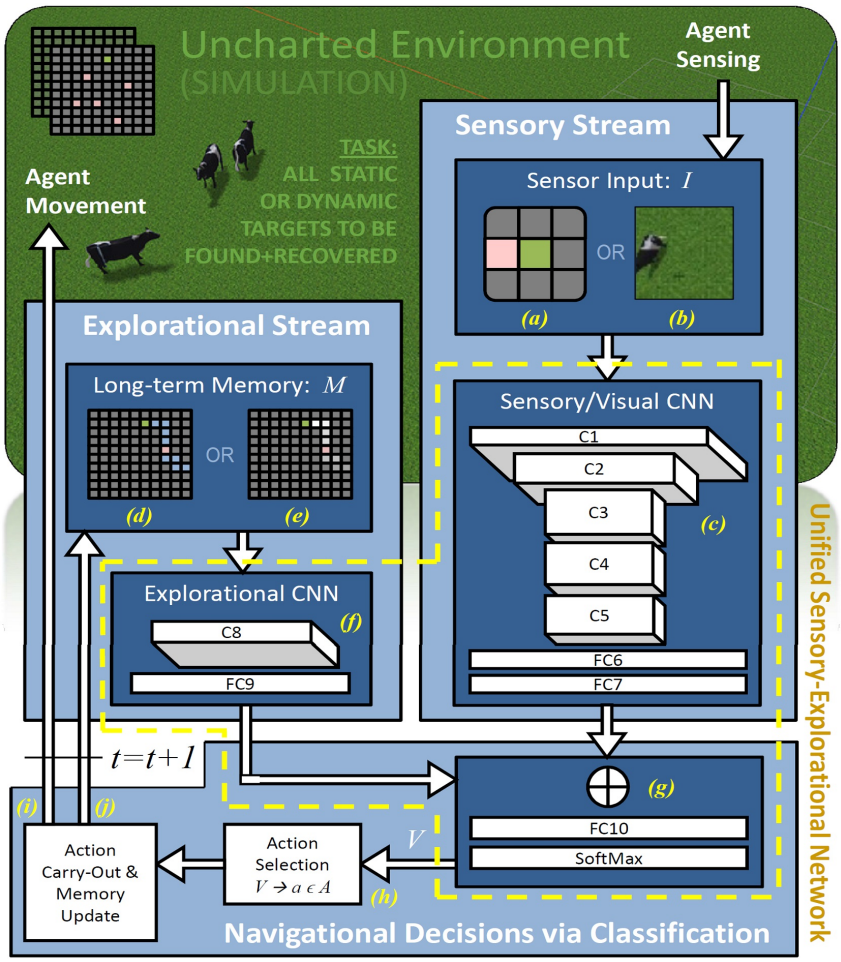


Figure 6.1: ***Unified Dual-Stream Deep Architecture for Search and Recovery Tasks.*** *The proposed design models the interpretation of sensory (tactical) and historic navigational (strategic) information within a single deep network (dotted yellow), which allows for unified back-propagation of navigation decision errors across both domains. Starting at the top right, the flow chart shows the agent's sensory input I, that is either* ***(a)*** *an abstracted, or* ***(b)*** *rendered, nearby environment sample. The input is processed via* ***(c)*** *a sensory/visual CNN utilising a basic AlexNet [171] design. In a second parallel stream, an evolving positional history memory M (holding either* ***(d)*** *spatial, or* ***(e)*** *spatio-temporal long-term information) is used as tensor input into* ***(f)*** *a network interpreting exploratory histories. Both streams are concatenated into* ***(g)*** *a shallow integration network that culminates in a softmax map towards a score vector output V over the possible navigational actions $a \in A$. During training, the entire deep network (dotted yellow) is optimised based on triples (I,M,V) using one-hot encoding of V and cross-entropy loss. During inference at time step t, the network receives a sensory input I and* ***(h)*** *selects the top-ranking navigational action (a) based on V, which is* ***(i)*** *performed and, in-turn,* ***(j)*** *the positional history M is updated. In order to initiate a next iteration time-step $t+1$, a new sensory I is sampled from the environment E closing the operational loop.*

Alternative approaches reside naturally in reinforcement learning, which is argued against in Section 2.4.2. The argument primarily focusses around an unsupervised learning phase being unnecessary; the positions of *simulated* targets are known and TSP solutions provide optimal decisions that can be trained against using the proposed classification architecture (see Figure 6.1). When however, the distribution of targets is unknown or challenging to accurately model (as is the case for real cattle targets), searching for reasonable exploration/exploitation policies using reinforcement learning makes intuitive sense[1]. A different approach could employ a particle filter; the agent iteratively refines estimates of target distribu-

[1] Implementing an unsupervised learning phase in reality may however be very challenging from an operational standpoint (e.g. UAV flight times are relatively low ∼10 minutes, thus requiring many batteries, flights would require constant human supervision, the search may not converge in reasonable time).

tion over multiple observations, with significant research existing in the related area of robot localisation, mapping and navigation [123, 64, 119]. The resulting distribution estimate however, still requires navigating towards visiting all targets. This complete process bears strong resemblance with the proposed architecture – target observation positions (explicitly recorded in long-term memory) implicitly indicate the distribution of other targets and actions are selected towards exploring those areas – except that these components are unified under a single recurrent classification pipeline.

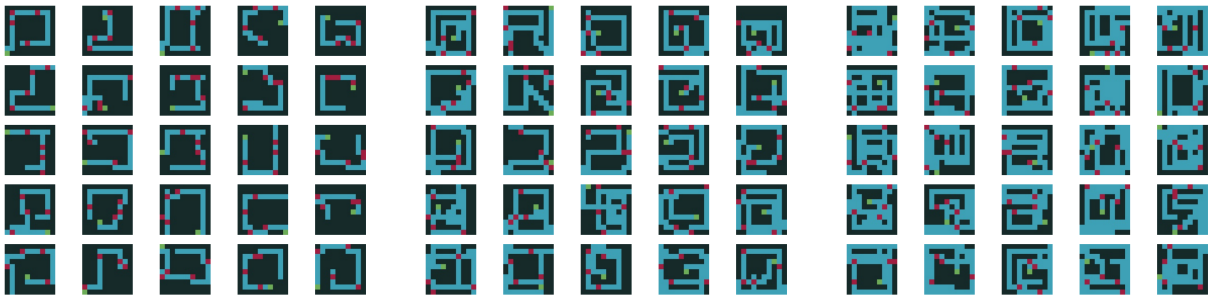
In order to focus on the core of the proposed methodology and to be able to operate on tractable computational grounds, real-world layouts are abstracted from here in the form of:

1. modelling time discretely,
2. representing space as a simple, discrete two-dimensional grid world and,
3. assuming grid-cell agent localisation to be resolved perfectly.

Despite the simplification, this abstraction is proven effective in real-world settings via the proof-of-concept experiments conducted in subsequent Chapter 7. Proposed as early as 1990 [300], grid worlds remain popular for exploring the viability of AI solutions today [211, 345]. In this chapter, agent-centric visual sampling – approximating a low-flying UAV with a downward-facing camera – is generated either from:

1. occupancy of local sub-grids of the world as content-independent abstractions, or
2. rendering from 3D Gazebo simulations [164, 345] for domain-specific, high-resolution images of particular target and environment content (e.g. grazing cattle herds). See Appendix A for the full implementation details on the simulation environment.

In the remainder of this chapter, Section 6.3 details formalisation of the problem mathematically, before describing training data synthesis of ground-truth actions in Section 6.4. Section 6.5 discusses the proposed solution and implementation, and subsequently, Section 6.6, describes experiments alongside quantitative findings. Finally, in Section 6.7 conclusions are drawn and future work indicated.



(a) $0 \leq \tau \leq 10$ *(b)* $20 \leq \tau \leq 30$ *(c)* $40 \leq \tau \leq 50$

Figure 6.2: ***Examples of Solved Testing Episodes.*** *Depiction of* 75 *example episodes where the agent (trained on the base case: Random (PT)) has recovered all targets at varying degrees of efficiency (avg.* 19 *steps for* 5 *targets over* 20*k samples – compared to lawn-mower pattern requiring* ≥ 41 *steps). (Pink): target positions uniformly randomly generated (specifying target distribution: P) at the beginning of an episode, (blue): past agent positions, (green): current/finishing agent position, and (black): unvisited grid positions. Instances are sorted (a) - (c) according to the difference in generated and optimal solution lengths* $\tau = |S| - |S_{GO}|$.

6.3 Problem Formalisation

6.3.1 Base Case – Static Target Recovery

For target recovery in a grid world E under discrete time $t_i \in \{t_0, t_1, ..., t_e\}$, let the environment E be defined as a rectangular 2D matrix with dimensions $w \times h$. The agent G and static targets $r^i \in R$ both have

discrete coordinates (x, y) within the map boundaries:

$$\begin{aligned} 0 \leq x < w,\ 0 \leq y < h, \\ \text{where } \ x, y \in \mathbb{Z}. \end{aligned} \tag{6.1}$$

Exactly $|R|$ targets are placed according to some unknown (possibly random) distribution P in this world. Note that G may take position anywhere, whilst multiple targets cannot occupy the same map position at any time-step:

$$\begin{aligned} \forall i, j \in |R|,\ t_k \in \{t_0, t_1, ..., t_e\} : r_x^i \neq r_x^j \wedge r_y^i \neq r_y^j, \\ \text{where } \ i \neq j \text{ and } i, j \in \mathbb{Z}. \end{aligned} \tag{6.2}$$

Possible agent actions $a \in A$ are the four possible navigation directions for non-boundary coordinates:

$$\begin{aligned} \text{forward } (-y) \\ \text{backward } (+y) \\ \text{left } (-x) \\ \text{right } (+x) \end{aligned} \tag{6.3}$$

or $A = \{f, b, l, r\}$, respectively for a top-left origin. Particular actions are inapplicable in the case that they would move the agent outside the map, e.g.:

$$\begin{aligned} &\text{if } \ G_x = G_y = 0, \\ &A|G = \{b, r\},\ \forall w, h > 0. \end{aligned} \tag{6.4}$$

Performing one particular action (e.g. "move right one unit", or $x := x + 1$) is defined to take one agent step in discrete time t. The agent is deemed to have recovered a target r^i iff:

$$G_x = r_x^i \ \text{ and } \ G_y = r_y^i \tag{6.5}$$

at some time-step t_j. A recovery solution S to an episode (i.e. visiting all $r^i \in R$) is defined as an ordered action sequence given initial agent coordinates (e.g. $S = [f, f, l, b, r, r, b]$, $G_x^{init} = G_y^{init} = 1$). An episode terminates in S once full target visitation is achieved, or in failure due to time expiry: $t > t_e$. The quality of a solution S is defined by its cardinality $|S|$ (time/steps required to visit all $r^i \in R$), the quality of exploratory agency is defined as the average solution quality (e.g. measured as target recovery rate) computed by scenario sampling given P.

Per time step, let the agent G be provided with two observable inputs:

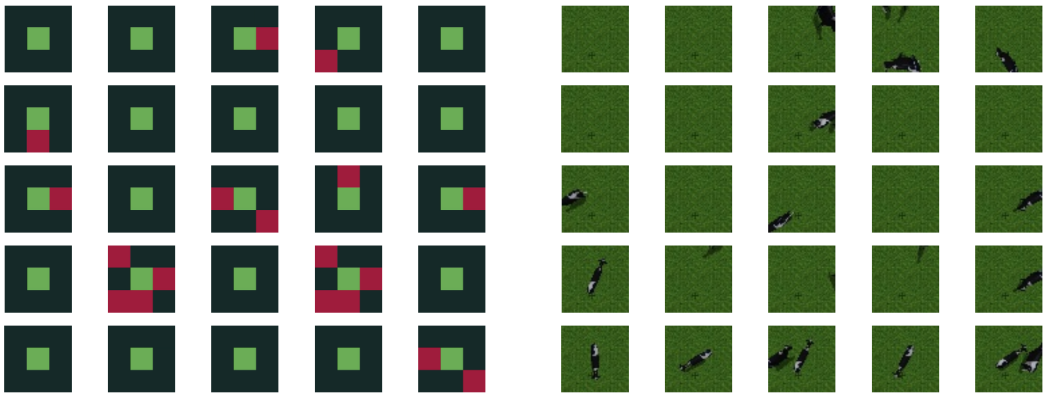
1. its own current position (G_x, G_y) in coordinates of E such that $G_x \in [0, w)$, $G_y \in [0, h)$, $G_x, G_y \in \mathbb{Z}$ and,
2. an agent-centric 3×3 grid image I resolving target occupancy in E topologically adjacent to G's location – in essence, neighbouring targets and unoccupied cells can be sensed, as depicted in Figure 6.1 ***(a)*** and Figure 6.3.

For the base case, this abstracts from realistic visual inputs (such as Figure 6.1 ***(b)***) and provides a content-independent problem isolation of tactical sensing from more fine-grained visual recognition tasks. The current agent position G_x, G_y is continuously recorded to keep a $w \times h$ spatial occupancy map M up-to-date, encoding for each position of world E:

1. if exploration has taken place *and*
2. if the agent is currently located there (see Figure 6.1 ***(d)***).

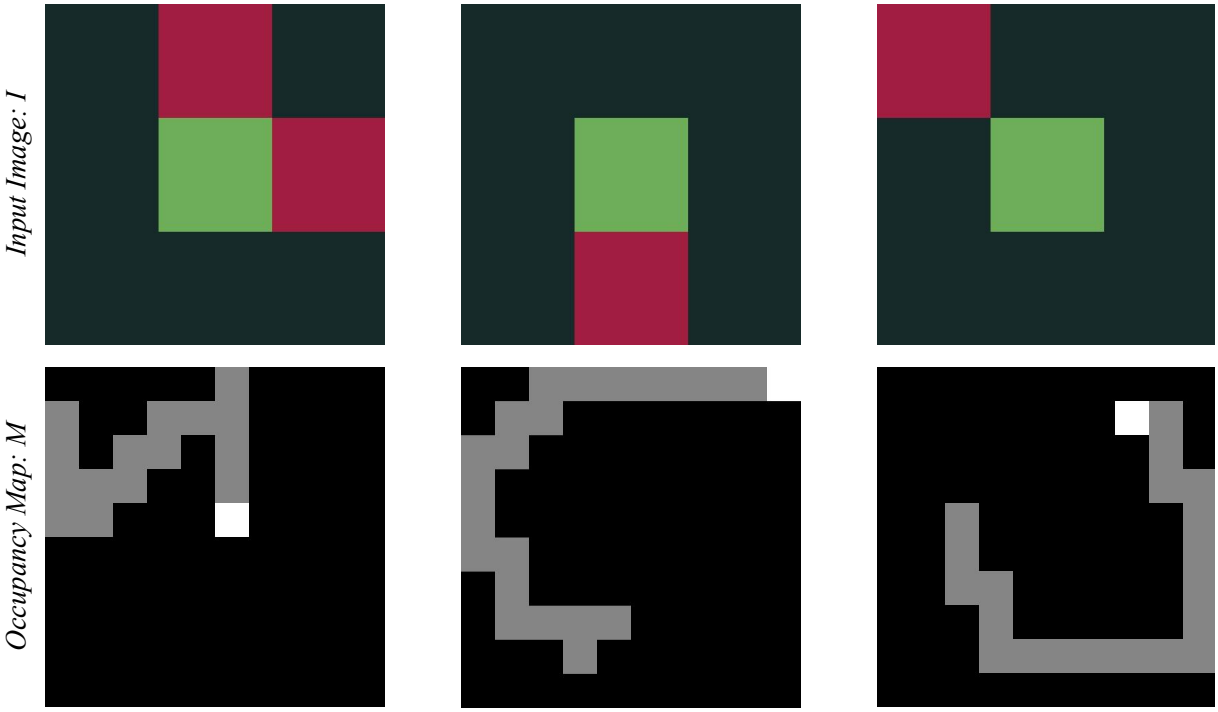
M in this form provides long-term external memory of positional history. Given this complete setup, the problem can be stated as:

> *How well can deep learning solutions learn how to optimise for fastest search and rescue of all targets given only samples of navigation decisions* (I, M, a)*?*



(a) 3×3 *pixel sensory abstractions.* *(b)* 50×50 *pixel simulation renderings.*

Figure 6.3: ***Agent-centric Visual Field Examples.*** *Examples of the two possible types of agent-observable visual input I of the environment from (a): sensory abstracted visual field or (b): Gazebo-rendered simulation environment (see Appendix A for implementation details on the simulation environment).*



(a) *Ground-truth action: r* ***(b)*** *Ground-truth action: b* ***(c)*** *Ground-truth action: f*

Figure 6.4: ***Ground-truth (I,M,a) Tuple Examples.*** *Examples of ground-truth* (I, M, a) *tuples for newly-generated episodes. Inputs consist of image I (top row): agent-centric visual field and M (bottom row): occupancy map of visited coordinates whilst output is a: corresponding ground-truth/optimal action given the inputs for some episode configuration, agent starting coordinates and target distribution P.*

6.3.2 Dynamic Case – Actively Moving Target Recovery

Here, the base case is extended for target motion over time – thus, distribution of targets P is now three-dimensional. Targets now apply individual actions $a_i \in A_{ext}$ at velocity $\frac{1}{s}u/t$ (spatial grid units per time-step) – whereas the agent displaces at velocity $1u/t$. Velocity with $s = 3$ is selected, yet $s > 1$ to make agent-target visitation eventually always possible. The action set is extended here to include action 'do nothing' denoted by n yielding:

$$A_{ext} = \{f, b, l, r, n\} \tag{6.6}$$

to permit optimised agent-target path intersection. Furthermore, single cell target occupation conditions given in Equation 6.2 are mandated $\forall t_i \in \{t_0, t_1, ..., t_e\}$. Architectural and temporal modifications are enacted on the agent's memory such that both agent and recovery locations are encoded spatio-temporally in the map (see Figure 6.1 ***(e)***), i.e. time annotations are memorised (see Figure 6.5). Specifically, changes to M involve an additional $w \times h$ dimension yielding new total size $w \times h \times 2$, where:

1. **Target location memorisation** ($M[:,:,0]$): upon $G_x = r_x^i \wedge G_y = r_y^i$ at some time-step t_j, the respective grid coordinate is marked with integer $\lambda = 50$. This encodes agent-target spatial and temporal visitation attributes.
2. **Agent location memorisation** ($M[:,:,1]$): G_x, G_y is marked with integer $\lambda = 50$ recording recent agent environment coordinate visitations.

Importantly, prior to these evaluations made at each t_i, every non-zero element of M is decremented by one. These modifications are made to account for individual target motion since: *(a)* we want to encode the uncertainty of target locations – knowledge of target whereabouts is solidified upon visitation and decays over time (due to random walk) according to Rayleigh's asymptotic approximation [255] and consequently *(b)*: memory of past environment agent positions now encodes less information regarding potential unvisited target coordinates. Put differently, these modifications allow the system to learn about the uncertainty of target locations by relating agent path over time and spatio-temporal memory – note that this information reveals properties of target motion beyond target placement statistics according to P.

(a) Target location memorisation $M[:,:,0]$ *(b) Agent location memorisation $M[:,:,1]$*

Figure 6.5: ***Temporal Occupancy Grid Map Example.*** *2D Sliced example of three-dimensional spatio-temporal occupancy map M at time t_i for a completed episode with environment E dimensions $w = h = 10$. For ease of visual representation, individual 2D slices $M[:,:,0]$ and $M[:,:,1]$ are encoded as grayscale images here (each element of M was linearly scaled from range $[0,\lambda]$ to $[0,255]$).*

6.4 Ground Truth Synthesis

To generate ground-truth solutions to new episodes – containing shortest routes visiting all targets – for the purpose of training data synthesis of appropriate navigation decisions (I, M, a), three strategies (detailed as follows) are employed. Since at this stage target positions within E are known, the problem is equivalent to the TSP, which has been well studied [129, 74, 88, 53, 95, 120, 141, 311, 285, 103, 152].

6.4.1 Fast Approximation: Closest Unvisited Target (CU)

Approximative solutions are generated fast by determining the position of the closest unvisited target $r^i \in R$, selecting an appropriate action $a \in A$ towards visiting r^i and repeating until all targets have been visited. This forms an approximation in a Nearest Neighbour (NN) fashion to any environment configuration irrespective of environment parameters w, h, or target set cardinality $|R|$. Note that this method may only sometimes produce optimality, as is reflected quantitatively in Table 6.1.

Action selection is determined by first finding the angle θ between the agent G and the closest unvisited target r^i using:

$$\theta = atan2(r^i_y - G_y, r^i_x - G_x). \tag{6.7}$$

Second, the action $a \in A$ is selected via $\frac{\pi}{2}$ intervals:

$$a = \begin{cases} rand(\{f, r\}), & \text{if } \theta = \frac{\pi}{4} \\ rand(\{b, r\}), & \text{if } \theta = -\frac{\pi}{4} \\ rand(\{b, l\}), & \text{if } \theta = -\frac{3\pi}{4} \\ rand(\{f, l\}), & \text{if } \theta = \frac{3\pi}{4} \\ f, & \text{if } \frac{\pi}{4} < \theta < \frac{3\pi}{4} \\ b, & \text{if } -\frac{3\pi}{4} < \theta < -\frac{\pi}{4} \\ l, & \text{if } \frac{3\pi}{4} < \theta < -\frac{3\pi}{4} \\ r, & \text{otherwise.} \end{cases} \tag{6.8}$$

Where $rand(X)$ randomly chooses an element $x \in X$.

6.4.2 Optimal Solution: Permutation of Targets (PT)

Any valid solution to an episode will visit all targets sequentially in some ordering. Globally-optimal solution(s) will therefore be some ordering on R that consist of the shortest number of moves (minimal $|S|$) given G^{init}_x, G^{init}_y and P. The number of possible orderings is factorially dependent on the cardinality of the target set, that is, within $O(|R|!)$ A particular target sequence is then fulfilled by selecting relative angle-appropriate actions (see equations 6.7 and 6.8) in a process akin to line rendering or rasterisation in images.

Target sequence orderings are generated via exhaustive tree search/growth. For the directed tree $T = (V, F)$, vertex attributes denote the target that node visits $v^i_{attr} = r$ for $r \in R$, $v \in V$, whilst edge attributes detail the optimal navigation action sequence connecting two targets determined via line rendering (e.g. $[f, f, l, b, r, r, b]$) using equations 6.7 and 6.8. The only exception is the root node v_0, which is defined to represent agent starting coordinates G^{init}_x, G^{init}_y within a particular episode. A child vertex and connecting edge for some v^i is added for all unvisited targets such that:

$$\forall v_i \in T(V), v_i \neq v_0 : \ deg^-(v_i) = 1. \tag{6.9}$$

A solution to the episode is formed by finding a sink vertex v_j with $deg^+(v_j) = 0$ and traversing upwards until the root source vertex v_0 is found with $deg^-(v_0) = 0$. During graph traversal, action sequence edge attributes are pushed onto a Last In First Out (LIFO) stack to form the final solution S. Optimality score for a solution is taken to be the number of actions, length of the action sequence or size of the queue $|S|$.

An intrinsic elegance of this solution is its independence in complexity from environmental dimensions $w \times h$, rendering it generalisable to higher-granularity environments (e.g. real-world robotic problems) with negligible runtime overhead. Whilst this approach guarantees optimality, it would be preferable to

use the alternative, aforementioned "Closest Unvisited Target" strategy (see section 6.4.1) for large $|R|$ due to exponentially-growing computational complexity (see figure 6.8).

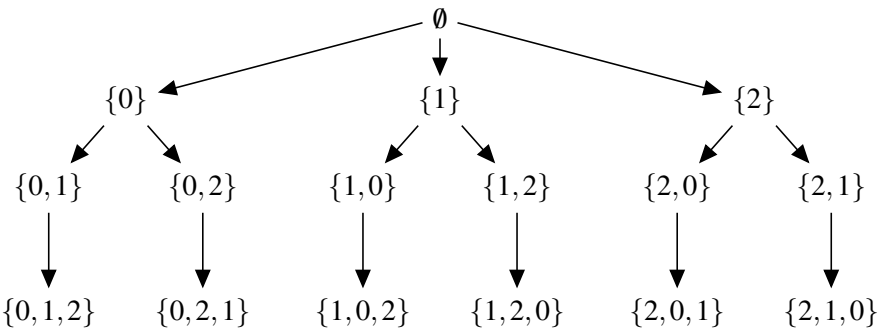


Figure 6.6: ***Target Orderings Graph.*** *All possible Permutations of Target (PT) orderings for $R = \{0,1,2\}$, $|R| = 3$. Edges represent Euclidean distance between vertex (target) coordinates in E. The root node $\emptyset$ represents the agent's starting coordinates within E: $G_x^{init} \in [0, w-1]$, $G_y^{init} \in [0, h-1]$ for some generated episode.*

6.4.3 Exhaustive Coordinate Search

As an alternative solution guaranteeing optimality, this approach exhaustively searches every possible coordinate within E. This comes at the obvious significant cost of computation time increase owing to solution complexity now being wholly dependant on environment dimension parameters (w,h). However, the benefit of this approach lies within its guarantee of eventual optimality with *no* knowledge of P (target spatial distributions). Tree growth formalisation is given as follows.

The tree $T = (V,F)$ is defined here as a directed, acyclic graph subject to the conditions:

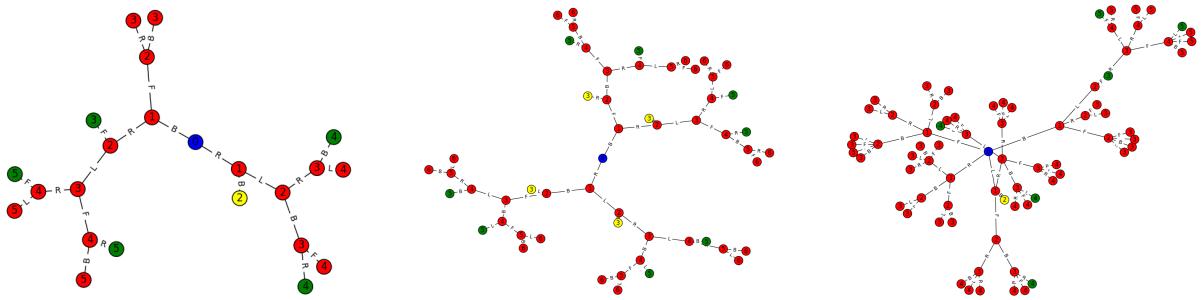
$$\begin{aligned} \underset{v_i \in T(V)}{\operatorname{argmax}}\ deg^+(v_i) = |A| = 4 \\ \text{and } \forall v_i \in T(V), v_i \neq v_0 :\ deg^-(v_i) = 1, \end{aligned} \tag{6.10}$$

where each vertex $v \in T(V)$ represents a grid coordinate $v_x, v_y \in E$. At the root node, v_0 directly represents the agent's starting coordinates G_x^{init}, G_y^{init}. Connecting edges $f \in T(F)$ thus denote a single legal action $a \in A$ (see Equation 6.4) connecting adjacent grid coordinates (graph vertices), therefore:

$$\begin{aligned} \forall uv \in T(F) : \sqrt{(u_x - v_x)^2 + (u_y - v_y)^2} = 1 \\ \text{where } u, v \in T(V). \end{aligned} \tag{6.11}$$

To bound computational costs (recursion depth) within reasonable time/memory constraints, a threshold is placed on paths with no new target visitation of length $\geq \phi_{max}$. This tree is grown recursively until a solution is found at a certain depth – directly encoding the number of moves required to solve the instance – which is not exceeded by other branches. If a better solutions is found by an alternative branch, it is established as the new maximal recursion depth.

As can be seen in Figure 6.7, graph complexity, even for small $w, h, |R|$ and ϕ_{max} is expansive. This solution was quickly therefore found to be computationally infeasible given reasonably natural environment-operational parameters (e.g. $w, h \geq 5$ and $|R| \geq 4$). Additionally, within the problem space here, in generating new ground-truth episodes, target locations are generated and thus known. As a result, the use of this solution is unnecessary, but useful in pure exploratory circumstances. Since the aforementioned PT strategy equally guarantees optimality, this ground-truth synthesis method was not brought forward for further experimentation or usage.



(a) $w = h = 2$, $G_x^{init} = G_y^{init} = 0$, $R = \{\{1,1\},\{1,0\}\}$, $|T(V)| = 23$, $\text{argmin}_{|s| \in |S|} = 2$ *and* $\phi_{max} = 3$.

(b) $w = h = 2$, $G_x^{init} = G_y^{init} = 0$, $R = \{\{0,1\},\{1,0\}\}$, $|T(V)| = 55$, $\text{argmin}_{|s| \in |S|} = 3$ *and* $\phi_{max} = 5$.

(c) $w = h = 3$, $G_x^{init} = G_y^{init} = 1$, $R = \{\{0,1\},\{0,2\}\}$, $|T(V)| = 76$, $\text{argmin}_{|s| \in |S|} = 2$ *and* $\phi_{max} = 3$.

Figure 6.7: ***Exhaustive Coordinate Search Graph Examples.*** *Various examples of the directed, acyclic graphs yielded by exhaustive coordinate search under varying parameters. Node colours are defined here as* ***(red)****: default node colour,* ***(blue)****: root node,* ***(green)*** *non-optimal solution to the problem and* ***(yellow)****: the optimal solution (least amount of moves). Vertex number attributes represent the current length of this solution (distance from the root node) whilst edge* $f \in T(F)$ *letter attributes denote* $a_i \in A$ *connecting two adjacent grid coordinates. Resulting complete graphs for:* ***(a)*** *a simple* 2×2 *environment size and maximum search depth* $\phi_{max} = 3$*,* ***(b)*** *increased search depth* $\phi_{max} = 5$ *and* ***(c)*** *larger environment size* $w = h = 3$*.*

6.5 Implementation

6.5.1 Recurrent Network Architecture

As previously introduced, Figure 6.1 illustrates and explains the end-to-end deep architecture used here. The reader can observe there that network output consists of a class membership score vector V with $|V| = |A|$ and $\sum_{v \in V} = 1$ as enforced by a final softmax-activated fully connected layer. In general, the model-selected action $a \in A$ is taken to be the maximal-likelihood value of V. Note that, if enacting a results in the agent moving outside of the environment boundaries, a random valid action is performed instead and equally, loop-detection may also alter action selection (see Figure 6.1 ***(h)***). Furthermore, note that the grid occupancy/visitation map M (see Figure 6.1 ***(e)***) intrinsically forms spatio-temporal memory for the agent with explicitly-defined architecture. This is as opposed to popular recurrent units such as GRUs [50] and LSTM [135] where the encoding and selection of temporal knowledge is implicit and hidden. Since all information of the path can contribute to learning information about distribution P and target movements, full and explicit long-term memory should intuitively perform well without such designs. Furthermore, since all training instances are modelled temporally independent/non-correlated, standard batch-based training can be employed here as opposed to resource-intensive back-propagation methods for recurrent architectures such as BPTT [327] and RTRL [261].

6.5.2 Infinite Loop Detection

An inherent property of model output (i.e. next-step action selection) is the potential for performing infinite agent loops. In their simplest forms, infinite loops are easily detectable via direct substring search into past actions (e.g. $\{f,b,f,b,...\}$, $\{r,b,l,f,...\}$). However, specific substring rules for more complex loop formations – consisting of many actions – are difficult to formulate and do not generalise well. Instead, loop detection here is primitively indicated in the event that the agent visits a particular grid coordinate α times (value $\alpha = 3$ is empirically set to minimise solution length).

Upon indication of a loop, action selection control (see Figure 6.1 ***(h)***) is given to an alternate algorithm, which in turn navigates the agent towards the closest preferred unvisited location in the occupancy map M for current G_x, G_y. This is achieved by examining occupancy map values surrounding the agent in a

$\beta_{init} = 1$ radius. Unvisited coordinates in that radius form an action voting table for action selection. If no unvisited location exists within that radius value β, it is incremented $\beta := \beta + 1$.

6.5.3 Training Setup

Synthetic training data produced by the selected ground-truth synthesis method is utilised to train the dual-stream DNN model defined previously. For training end-to-end, a single instance (I, M, a) consists of inputs: agent-centric visual input I and occupancy grid map M, whilst output is an one-hot action-class vector encoding ground-truth action a used for back-propagation. To verify pure ground-truth classification performance, $k = 10$-fold cross validation is performed over the respective experimental dataset. At each fold k_i, training is performed for 50 epochs over the partitioned training set with weights initialised randomly from a truncated normal distribution and a batch size of 64. Parameters are optimised via categorical cross-entropy loss using SGD with momentum [250] and fixed learning rate $e = 0.001$. Mean and standard deviation cross-validated classification accuracies for each experiment (where applicable) are given in Table 6.1: ***(h)***.

6.5.4 Baseline Algorithms

Three baseline methodologies are implemented here to compare and assess the performance of the proposed dual-stream, unified deep learning approach against:

Naïve Solution (NS)

As a simple, naïve strategy to provide a primitive baseline, an algorithm is employed whereby if *(1)*: an unvisited target is currently present within the visual field (verified by examining M), select an angle-appropriate action towards it using Equation 6.8. Otherwise, *(2)*: navigate towards a next unvisited location using the same voting strategy given in Section 6.5.2 (otherwise used to break loops), and repeat *(1)* until the episode terminates from time expiry or complete target visitation.

Deep Recurrent Q-Network (DRQN)

Also implemented is an off-the-shelf DRQN [126] for comparison employing a 1000-time-step strong experience replay buffer. This baseline is established to validate the hypothesis that exploratory agency benefits from known solution samples and the inclusion of structured, map-based long-term memory. Visualisation of the environment was modified with a white border surrounding the map used to create the partial visual input I given to an agent. This serves to provide agent knowledge of the environment boundaries – which is an implicit property of the occupancy grid map M's architecture for the method described here. Additionally, target visitation is visually signified to the agent who can then no longer receive reward from that target.

Split Stream Network (SSN)

To validate the motivation of the proposed dual-stream approach – that exploratory agency benefits from information exchange between sensor and positional history inputs trained under a single architecture – the architecture is split here during back and forward propagation. That is, resulting split stream networks 1 and 2 optimise on (I, a) and (M, a) using network architectures ***(c)*** and ***(f)*** as given in Figure 6.1, respectively. Network outputs are element-wise summated and normalised to yield the final action-class score vector used for action selection.

			(a) Loop Detected (%)	(b) >1 Loops Detected (%)	(c) >100 Moves (%)	(d) Optimal Solution (%)	(e) ≤ 10 Difference (%)	(f) $t > t_e$ Moves (%)	(g) Target Recovery Rate ($^d/_t$)	(h) 10-fold CV mAP (%)
6.6.2	**Baselines for Static Recovery** (random targets)	NS	-	-	8.13	0.34	5.96	0	0.260±0.113	-
		DRQN [126]	-	-	58.09	0.02	2.18	45.56	0.237±0.092	-
		SSN (PT)	56.31	63.16	23.52	0.66	3.89	0.01	0.217±0.112	68.95±0.256
6.6.1 + 6.6.4	**Learning Static Recovery** (agent location memorised)	Random (CU)	37.49	53.01	9.64	2.21	13.76	0	0.262±0.113	67.84±0.237
		Random (PT)	32.45	46.78	7.57	2.85	15.72	0	**0.265±0.114**[1]	71.15±0.233
		Fixed Grid	0.94	100	0.76	99.06	99.06	0.63	0.337±0.054	91.47±0.696
		Equidistant	54	82.01	19.63	21.78	45.75	0.91	0.333±0.109	83.66±0.182
		Gaussian	24.46	30.17	0.58	39.99	78.61	0.07	0.616±0.143	75.81±0.515
6.6.3	(simulation)	Random (+S)	36.73	44.94	6.88	6.53	35.72	0.164	0.289±0.109	73.97±0.175
6.6.5	(agent+recovery locations memorised)	Random (+M)	70.16	78.9	25.55	2.08	12.67	0.59	0.261±0.115	70.17±0.276
		Equidistant (+M)	47.59	74.79	7.41	30.25	60.87	0.16	**0.380±0.085**[2]	84.99±0.224
		Gaussian (+M)	23.52	63.6	3.28	45.46	78.28	0.15	**0.622±0.141**[3]	77.30±0.429
6.6.6	**Learning Dynamic Recovery**	Random Walk	-	-	79.17	0.08	0.93	12.23	0.155±0.176	60.91±0.418
		Herd Walk	-	-	77.18	0.88	3.18	21.39	0.225±0.128	62.33±0.273
		Herd Walk (+S)	-	-	61.35	0.12	2.90	5.67	**0.203±0.098**[4]	63.26±0.181

*Table 6.1: **Performance Overview.** Row sections in the table group results of the proposed approach across three task categories, and against three baselines, respectively: 6.6.2: **Baselines** for recovery of static, randomly placed targets all outperformed by [1]the dual-stream approach; 6.6.1+6.6.4: **Static Recovery** attempting to learn recovery under various spatial target distributions P; 6.6.3: Using **Gazebo Simulations** (+S) instead of abstracted sensing; 6.6.5: Additionally **Memorising Recovery Locations** instead of only agent locations, shows superior results for all [2,3]non-random target placements; 6.6.6: **Dynamic Recovery** attempting to learn recovery under target motion including [4]Gazebo simulated herd dynamics. Columns hold the following values: **(a)**: % of episodes where a loop was detected; and **(b)**: of those cases, % of episodes where another, subsequent loop was detected; **(c)**: % of episodes where the model required more steps than simply exhaustively exploring every environment position (for fixed $w = h = 10$); **(d)**: % of episodes where the generated solution length was equal to the optimum; and **(e)**: % of instances where the model generated a solution less than 10 moves longer than the optimum; **(f)**: % of instances where time expiry (model failure) occurred with $t_e = 300$; **(g)**: $\mu \pm \sigma$ target discovery rates (no. of target recoveries per time-step); **(h)**: 10-fold cross validated ground-truth classification mean average precision $mAP \pm \sigma$.*

6.6 Experiments

Importantly, throughout experiments and indeed other places in this chapter, 'random' numbers are generated using the P-RNG Mersenne Twister algorithm [208] but are referred to as random. The number of targets is fixed to $|R| = 5$, environment E dimensions are set to $w = h = 10$ and unless specified otherwise, 20,000-episode strong training datasets are synthesised per experiment containing approximately $400,000 \times$ instances of (I, M, a) data tuples. Note also that the agent's initial position is randomly generated at the beginning of every episode $G_x^{init} \in [0, w-1], G_y^{init} \in [0, h-1]$. Targets remain static throughout episodes (solving the base case given in Section 6.3.1, where distribution P is two-dimensional) in all experiments up until those given in Section 6.6.6. For testing, 20,000 newly generated episodes are presented to the trained models to solve, model recovery performance is then measured and reported. Results for all experiments are given above in Table 6.1 and evaluated in detail over the following subsections.

6.6.1 Episode and Ground Truth Generation

In this experiment, the two aforementioned ground-truth synthesis strategies (see Section 6.4) are compared and contrasted – that is, Closest Unvisited Target (CU) and Permutation of Targets (PT) – for training data generation utilised for subsequent model training. Figure 6.8 illustrates comparative results on newly-generated episodes with fixed environment dimensions $w = h = 10$ and random target

distribution P. As theorised for PT, the time required to solve episodes increases factorially with the number of targets, i.e. $O(|R|!)$, whilst CU requires negligible linear time (see Figure 6.8: *(middle)*). The trade-off being that as $|R| \longrightarrow \infty$, the average difference in generated solution lengths diverge (see Figure 6.8: *(left)*) – since PT guarantees optimality. Yet, as illustrated in Figure 6.8: *(right)*, employing the CU strategy yields optimality in the majority of instances ($\sim$60%). Comparison of both training synthesis strategies against validation episodes demonstrably proves that PT prevails (see both Table 6.1 and Figure 6.9) in learning the conditions for optimal decisions given random target distribution. Thus, the PT approach is employed for episode solution synthesis for all other experiments and this result (i.e. using PT) is established as the base case [1]benchmark for static, uniformly randomly distributed targets.

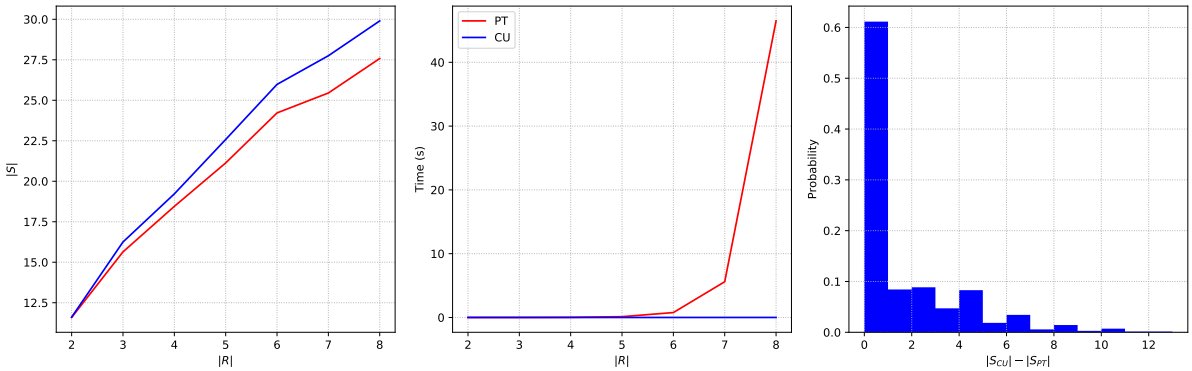


Figure 6.8: ***Ground-truth Synthesis – Comparison of Strategies CU vs PT.*** *Comparison of ground-truth synthesis methods for newly randomly generated episodes. At each value of $|R|$, 100 additional episodes were generated and solved. (Left): average ground-truth solution quality $|S|$ for generated solutions against the number of targets $|R|$. (Middle): average time required for solution generation against the number of targets. (Right): normalised histogram for the difference in generated solution length between CU and PT for 100 instances for each $|R| \in [2,8]$.*

6.6.2 Baseline Comparison

These experiments evaluate the three implemented baseline algorithms (see Section 6.5.4) against the proposed dual-stream architecture. For each algorithm, targets are uniformly randomly spatially distributed within E. Resulting performance statistics illustrate under-performance in all aspects in comparison to the aforementioned [1]benchmark result. The employed naïve approach (NS) yields highest baseline performance and is accordingly shown for comparison in Figure 6.9 against other benchmark results achieved by the solution proposed here. Off-the-shelf DRQN [126] demonstrably performs poorly despite access to experiences containing decisions leading to reward. This occurs arguably as a result of the agent having no global notion of current environment localisation necessary to locate possible future reward in an overall context. Splitting inputs into separately-trained neural network streams (SSN) is also shown to yield poor performance since model exposure to any single input (I or M) alone is insufficient in producing optimal reasoning (the problem becomes increasingly partially-observable). This being said, knowledge of occupancy grid M (encoding G_x, G_y, visited cells, etc.) inherently encodes more information on E than I, which is reflected in findings of significantly higher mAP classification accuracy in the network optimising on (M,a) alone. Additionally, performing end-to-end training on two separate networks to address this imbalance attributes significant additional computation over the proposed dual-stream network architecture.

6.6.3 Simulated Full Visual Input (+S)

To this point, sensory environment observability has been encoded by small 3×3 sensory abstractions. To simulate a more realistic target location recovery assignment – where visual target detection is no

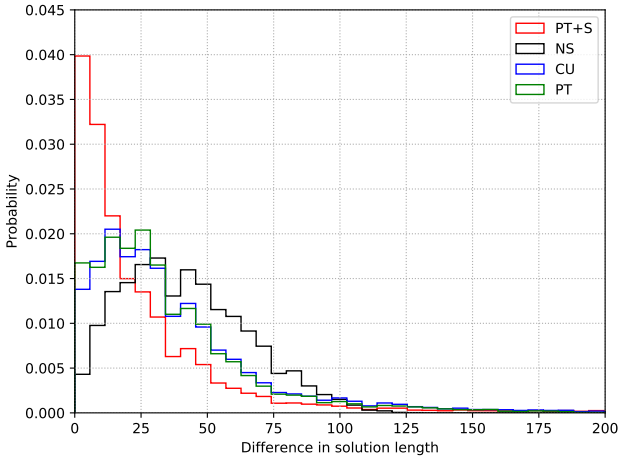


Figure 6.9: ***Model Solution Length Difference.*** *Normalised histogram of the difference in generated solution length using models trained with Closest Unvisited (CU) and Permutation of Targets (PT) ground-truth synthesis strategies against the global optimum over* $20,000$ *test instances. Distributions when using the naïve solution (NS) and Gazebo simulator [164] (PT+S) are also included. See Table 6.1 for further detailed performance statistics.*

longer trivial – the task of visual outdoor farming census related to [12]; discovering cattle positions within a field using a quadrotor UAV is introduced. The task is simulated utilising randomly-oriented 3D cattle models within the Gazebo robot simulation environment [164] (see Figure 6.10: *(top)*). The simulation environment is used to directly replace the agent's sensory field I with a 50×50 pixel image (see Figure 6.10: *(bottom)*) for a 100° FoV camera, whilst M remains identical in architecture and operation. The UAV is flown in discrete 2m increments within the horizontal xy-plane at fixed height $z = 3.5$m.

The experimental setup remains identical to the benchmark case with agent and cattle targets uniformly randomly spatially distributed. Whilst obtained results (see Table 6.1 and Figure 6.9) indicate an advantage for the simulator case, these results are not directly comparable within the setup used here, since environment observability improves in a three dimensional projection beyond 2D gridding given the camera FoV (see Figure 6.10: *(bottom)*), object shadows, etc. Whilst efforts could have been made to strictly enforce agent visibility to a 1 grid ground cell radius (as for the grid world), this would have failed to model real scenarios well since visual artefacts (e.g. shadows) are present in real-world sensing. Note that the employed visual processing CNN architecture (AlexNet [171]) clearly demonstrates that it can recover and utilise additional visual information found in such realistic robotic scenarios (see Table 6.1, Section 6.6.3).

6.6.4 Learning Static Recovery under Spatial Distributions

Experiments to this point have dealt with fully random uniform target distributions. Here it is shown that simpler and arguably more realistic spatial distributions can also be learnt (naturally, far more effectively) under the solution proposed here. Three additional distributions are employed as follows:

Fixed Grid

As a baseline distribution case, targets are distributed in a fixed, static grid across episodes (spatial distribution P is known completely), whilst agent position initialisation varies as before. Since there are many orders of magnitude less possible environment initialisation configurations, $wh - |R| = (10 \times 10) - 5 = 95$ – since only the agent position is varied per-episode – one would expect highly improved model

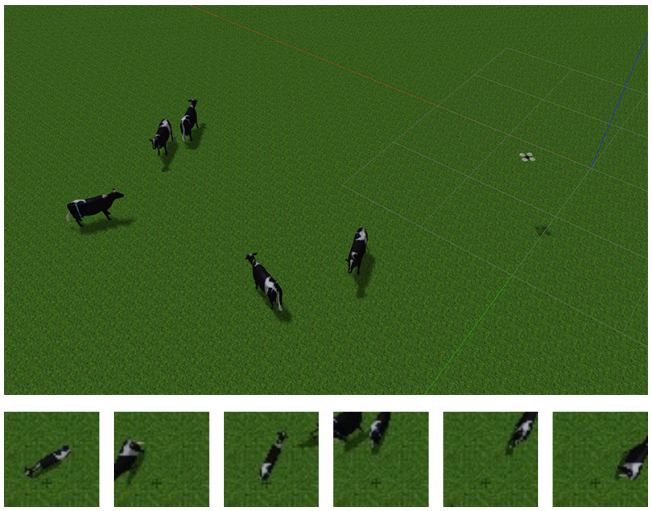


Figure 6.10: ***Visual Simulation Environment.*** *(Top): overview of the Gazebo-powered [164] simulation (+S) environment for a randomly generated static episode. (Bottom): example* 50×50 *pixel images from the simulated downward facing UAV camera provided as input to the agent.*

performance, which indeed turns out to be the case both in pure ground-truth classification accuracy and online model performance (see Table 6.1, row 'Fixed Grid').

Equidistant Grid

Targets are distributed spatially in a grid that satisfies target-target NN equidistance (Voronoi-like distribution) – a property that is observed in real cattle-herd distributions [301]. The position of the grid in E is varied randomly per-episode and inter-target NN spacing satisfies 2 grid-unit spacing. Since the grid is fixed in shape across episodes, positional knowledge of just two targets provides full information of remaining target positions. Ergo, the problem is then vastly less complex than the fully random case, which is reflected in strong model performance on this distribution (see Table 6.1, rows 'Equidistant').

Gaussian

Target positions are sampled from a two-dimensional Gaussian with constant parameters $\mu_x = 3$, $\mu_y = 5$ and $\sigma_x = \sigma_y = 1$ at the beginning of each episode. Values are sampled from the distribution until unoccupied environment coordinates are generated. Results demonstrably show strong model performance in efficiently discovering target positions – relatable to an ability to effectively learn the distributional parameters of the underlying 2D Gaussian P (see Table 6.1, rows 'Gaussian').

6.6.5 Memorising Recovery Locations (+M)

In these experiments, the visitation map M is modified such that target locations are marked upon agent visitation. This small implementation detail leads to improved performance upon all tested, non-random target distributions, that is 'Equidistant' and 'Gaussian'. These improvements occur intuitively; previously discovered target positions directly encode information about a non-random P. This is demonstrably not the case for fully random distributions – knowledge of randomly positioned target r_i reveals no information about the position of r_j (where $i \neq j$), and thus, yields no increase (i.e. slight decrease) in performance (see Table 6.1, compare rows 4, 5 and 10).

6.6.6 Learning Dynamic Multi-Target Recovery

Thus far, targets have remained static throughout episodes. Here, the implementation is extended to include individual target motion in two forms (a): random walk and (b): herd-inspired motion. Formalisation of this, now dynamic recovery problem, is given in Section 6.3.2. Note also that infinite loop detection – applicable to the static recovery task – is disabled in this context.

Random Walk

Targets implement random walk via random action selection $rand(A)$ with a fixed velocity of $\frac{1}{3}u/t$ (one grid unit every 3 time-steps). To determine globally-optimal solutions towards training data generation, $\varepsilon = t_e = 300$ random actions are pre-determined for each target such that Equation 6.2 – mandating single grid cell target occupancy – remains satisfied for every time-step $t_i \in [0, \varepsilon]$. The yielded matrix containing target coordinates over time is then exhaustively searched for each possible combination of future target visitation orderings towards finding the shortest solution using a full 'lookahead' extension of the PT solver.

Herd-like Motion

Looking towards more realistic scenarios closer to a robotic UAV application in 'smart farming', cattle herd-inspired distribution initialisation and motion is applied. Targets are initially distributed in an equidistant grid (as for Section 6.6.4) with random position in E. Herd-inspired motion is implemented with an overall group direction $d_i \in A$ randomly selected at the beginning of each episode. Direction d_i is applied to each target at velocity $\frac{1}{3}u/t$ with a 10% per-individual likelihood that they instead perform a random action $rand(A_{ext})$ excluding in-axis directions for d_i (e.g. if $d_i = r, A_{ext} = \{f, b, n\}$).

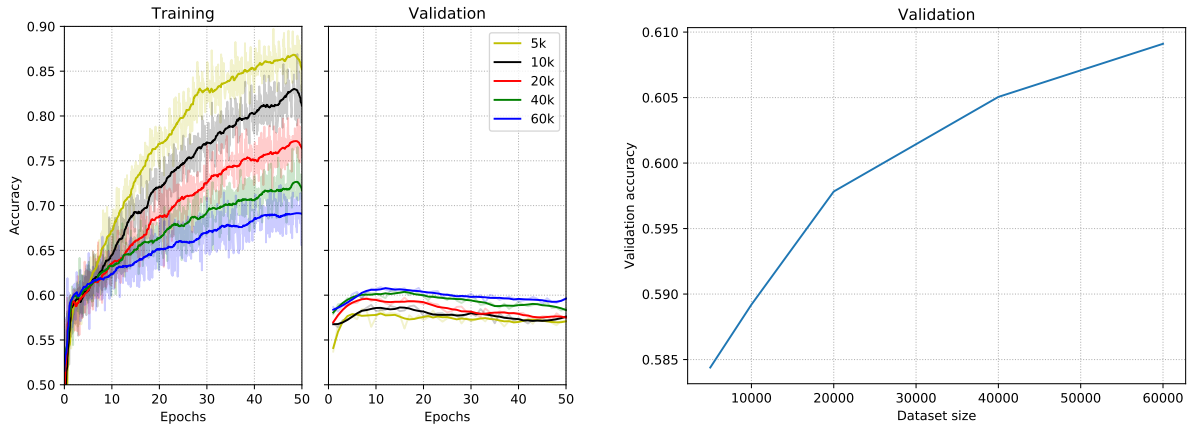
This motion behaviour approximation is supported by literature observing collective dynamics for grazing cattle herds [353]. Finally, upon reaching the boundaries of E, a new group direction d_{i+1} is randomly selected where $d_i \neq d_{i+1}$. Identically to random walk, $\varepsilon = t_e = 300$ individual motion actions are predetermined and solved for optimally via the full 'lookahead' extension of the PT solver.

Results Discussion

Following network training on the random walk experiment, the so far employed $20,000$-episode dataset cardinality was found to be insufficient for optimal performance leading to significant model over-fitting (see Figure 6.11a). This observation directly illustrates the significant increase in problem complexity introduced by individual target motion. Increasingly larger datasets improving validation accuracy were thus synthesised at the cost of overall computation time. However, reasonable synthesis, training and evaluation times were quickly exceeded – here it was opted to empirically limit dynamic experiment datasets to $60,000$ episodes as a reasonable accuracy versus time trade-off. Findings suggest that there is more validation accuracy to be gained – albeit with diminishing returns – by providing even larger sets of episode datasets (see Figure 6.11a).

Quantitative ground-truth classification and online model performance statistics given in Table 6.1 demonstrably indicate the capability of the proposed unified network to generalise well to the case of individual target dynamics. In particular, herd-inspired motion generally improves search and recovery capabilities (see Table 6.1, compare rows 13 vs 14 and 15). Finally, experimental outcomes to this point culminate in the last, highlighted experiment: 'Herd Walk (+S)', combining target dynamics enacting herd-like motion *and* visual sensory input rendered via the cattle-census simulation environment (see Section

6.6.3 and Figure 6.10). As for the static case, increased environment observability is observed and reduced partiality in view compared to the 2D case. Yet, the experiment clearly validates the employed dual-stream, single network architecture in yielding competitive exploratory decisions whilst processing higher resolution, complex visual imagery in unity with spatio-temporal navigational memory.



(a) Training and validation set accuracy vs. training steps for different dataset cardinalities

(b) Dataset cardinality or size vs. final validation accuracy

Figure 6.11: ***Training Evolution of Classification Accuracy for Dynamic Targets.*** *Classification accuracy for training and validation sets over 50 epochs over optimal generated target motion datasets (implementing random walk) with various numbers of thousands of episodes used for training. Refer back to Table 6.1, bottom row for quantitative findings. Signals in (a) have been smoothed using the Savitzky-Golay filter [275].*

6.7 Chapter Conclusion

This chapter demonstrates that recovery tasks can be effectively modelled by combining visually-motivated sensing and map-based positional histories under a single deep classification architecture, comprising the combined optimisation of both information streams in unity. This approach is shown to outperform the various tested baselines including split stream optimisation; justifying the employed dual-stream architecture. The proposed architectural choices demonstrably generalise well to a wide range of scenarios, target distributions and dynamism given training data comprised of good or optimal decision strategies yielded from travelling salesman solutions or otherwise. Concluding this part of the thesis dealing with environment exploration, it has been shown that costly unsupervised methods can be circumvented; efficient exploration strategies can be learnt from example under a single architecture. Work to this point now segues naturally into the following part detailing real experiments, combining local individual identification and exploratory agency on a real cattle herd.

Supplementary Information

Publication associated with this chapter:

1. W. Andrew, C. Greatwood and T. Burghardt. Deep Learning for Exploration and Recovery of Uncharted and Dynamic Targets from UAV-like Vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1124-1131, 2018. https://doi.org/10.1109/IROS.2018.8593751.

The source code, dataset and video accompanying the publication is available at:

- Source code:

  - https://data.bris.ac.uk/data/dataset/2zot65rxlmgqq23au92qwkaa3x
  - https://github.com/CWOA/gtrf
- Video:
  - https://vimeo.com/280747562

Chapter 7

Proof-of-Concept: Real-World Herd Individual Identification

7.1 Chapter Overview

This chapter presents the culmination of this thesis in operating a series of experiments performing live and online identity recovery of a small $|R| = 17$-strong cattle herd[1]. This is all performed on-board a UAV-based agent with limited available computational resources, payload size and weight restrictions as well as overall flight time constraints. Within this task, the viability of models exercising effective exploratory strategies in reality is assessed, validating the transition from extensive synthetic and simulated cases presented in Chapter 6 to real quadrotor operation[2] within a significantly larger exploratory domain. Furthermore, components performing iterative and integrative individual identification (Chapter 4) are validated in a live setting, strengthening the proposal that this is indeed an effective strategy for object classes requiring fine-grained identification – certainly across the small population assessed in this chapter. Over the preliminary set of experiments conducted in this chapter, results provide a proof-of-concept in suggesting this approach to be viable, whilst highlighting points for experiment and algorithmic improvement given future work.

This chapter is organised into: firstly, providing details on the hardware setup and respective choices used in this chapter (Section 7.2), followed by a description of the dataset acquisition, labelling and augmentation process used here to train relevant architectures in Section 7.3. Third, specific algorithmic and software implementation choices are given in Section 7.4, primarily discussing the modifications required to transition from simulated to real inputs. Fourth, Section 7.5 discusses the extensive preparation details required for the realisation of experiments involving the flight of an autonomous UAV agent. Next, conducted experiments are described in Section 7.6 alongside results and discussion points with finally, concluding remarks for this chapter provided in Section 7.7.

7.2 Hardware

This section provides details on the hardware setup used here across experiments involving a real UAV-based agent. The complete setup, along with relevant component interactions, is depicted in Figure 7.1 and consists of the UAV flight platform itself with many components housed on-board the platform and remote devices that control the operation of the aircraft. Wireless communications bridge the gap between the basestation and flight platform. However, all inputs and autonomous control are processed and issued on-board the UAV – the remote devices primarily act in a supervisory role. The flight platform and corresponding components housed on-board the UAV are illustrated in Figure 7.2.

[1] Special mentions of thanks go to Dr Colin Greatwood for piloting the UAV throughout this chapter in acquiring the dataset and supervising during autonomous flight experiments.

[2] Note that for every UAV flight, all CAA regulations for drones [16] were abided by.

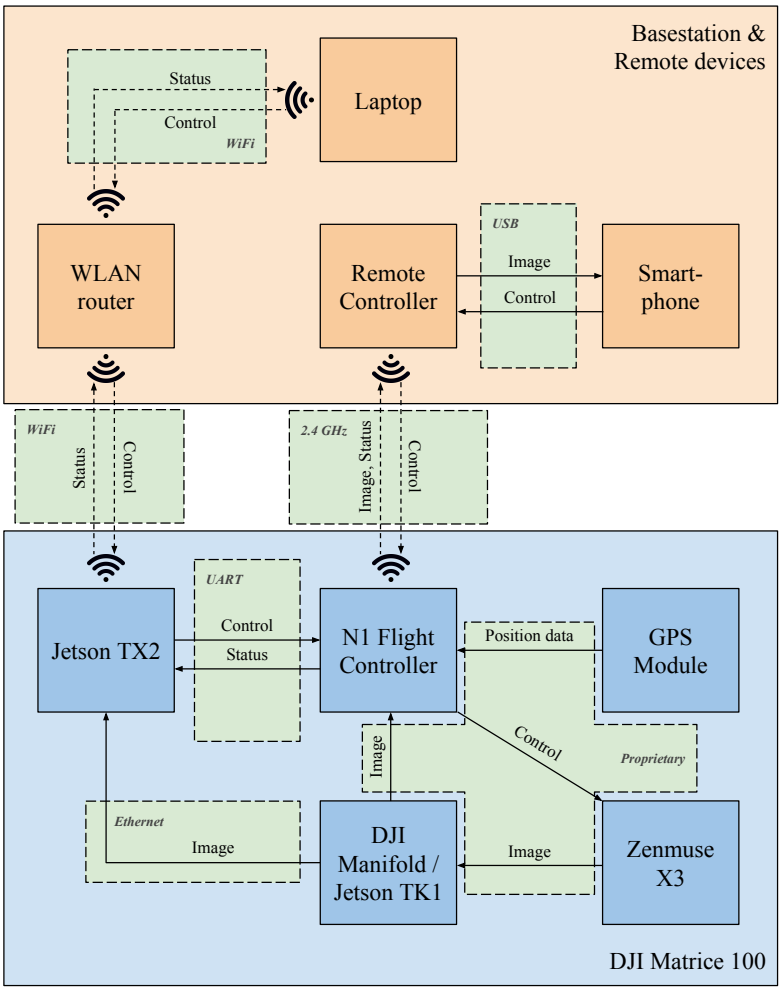


Figure 7.1: ***Full UAV Hardware Architecture.*** *Full UAV hardware architecture with labelled components and interactions on-board the flight platform itself (DJI Matrice 100) and its wireless interactions with remote devices exercising ultimate supervisory control. Communication interfaces between individual components (in green) are also given. Fully manual aircraft control is completed by the remote controller where a live camera feed is visible on an attached smart device. All programmatic commands are issued via ROS-based API calls over a serial connection between the Jetson TX2 (the primary on-board computer performing ANN model inference) and the N1 flight controller. Control of this form is initiated remotely via SSH over a WiFi connection and monitored live via ROS. Presence of a second on-board computer (the DJI Manifold) is necessitated by proprietary drivers guarding decoding live imagery from the Zenmuse X3 camera and gimbal system. The image is thus continually acquired, decoded and communicated via ROS to the Jetson TX2 by a wired Ethernet connection introducing* ~ 0.4*s latency.*

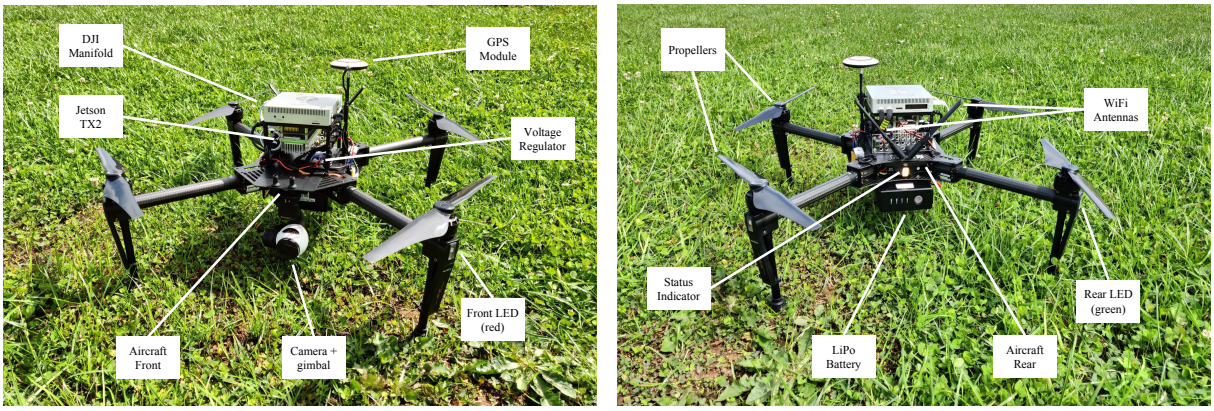
7.2.1 UAV Flight Platform

The DJI Matrice 100 – often abbreviated to M100 – is a configurable, performance quadrotor UAV specifically designed and marketed towards developers.[3] Available at a relatively low cost (~£3000), the M100 is a complete, all-in-one platform, requiring minimal assembly out of the box (full vehicle specifications are given in Table 7.1). Crucially, the on-board DJI N1 flight controller can be interacted with with a ROS-based API[4] including commands to fulfil relative target positions, autonomously take-off, land and more[5]. Importantly for use in this thesis, the platform has flight capability for up to 1kg of

[3] DJI Matrice 100: https://www.dji.com/matrice100

[4] DJI SDK repository: https://github.com/dji-sdk/Onboard-SDK-ROS and https://github.com/dji-sdk/Onboard-SDK

[5] Full list of supported ROS commands: http://wiki.ros.org/dji_sdk



(a) Aircraft front *(b) Aircraft rear*

Figure 7.2: ***Finalised UAV Platform.*** *Front and rear images of the complete DJI Matrice 100 UAV flight platform having just landed from test flights with selected individual components labelled. In flight, bright LEDs indicate the front and rear of the aircraft clearly to the operator whilst the status indicator LED at the rear of the aircraft displays battery level warnings, satisfactory GPS coverage, compass calibration requirements and more. Currently the status indicator is reporting in yellow that the controller is not connected. Additional LEDs on the LiPo battery itself indicate remaining charge in $^1/_4$ intervals.*

payload – sufficient for the components required here (e.g. camera, on-board computers) – at the cost of decreased flight time. Structurally, the main body features modular expansion bays and racks to house additional such components. As a result of these features, the use of the M100 is popular in literature; with work specifically found in autonomous pylon inspection [142], vision-based autonomous landing on moving platforms [57], wilderness Search and Rescue (SAR) [341] and more [182].

Parameter	Value
Maximum Speed	17m/s (no payload, no wind)
Hovering Time	13 min (1kg payload)
Maximum Takeoff Weight	3.6kg
Maximum Wind Resistance	10m/s
Maximum Speed of Ascent	5m/s
Maximum Speed of Descent	4m/s

Table 7.1: ***UAV Specifications.*** *DJI Matrice 100 advertised flight platform specifications.*[6]

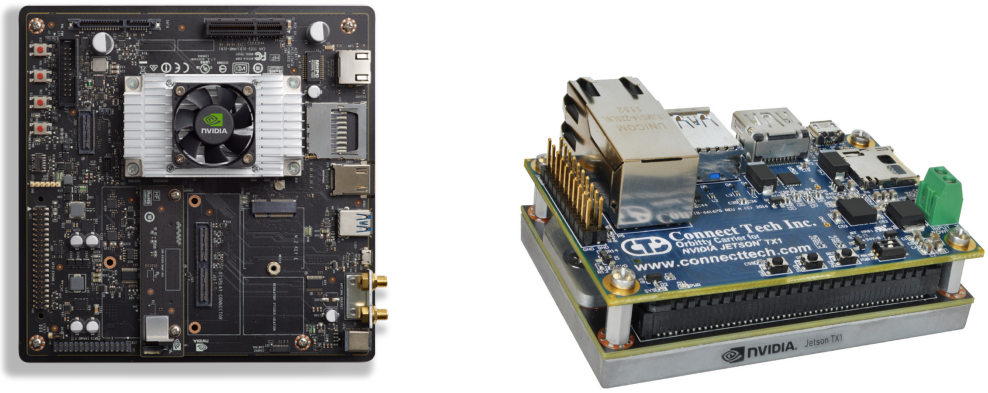
On-board Processing Devices

The justification of the decision to perform visual processing on-board the flight platform resides in several core contributing factors:

- ***Latency***: the visual processing of imagery on-board alleviates the fluctuating latency cost involved in wireless transmission of an image to a remote processing station and its subsequent response.
- ***Dependency & Range***: dependencies on a remote processing station limit the ease of deployability in new environments and introduce range constraints for wireless data transmission.
- ***On-board Embedded System (OES) GPU Compute Capabilities***: contemporary low-cost devices (as used in this thesis) are sufficiently capable of GPU-based computational load and, are physically small and lightweight whilst exhibiting minimal power consumption.

[6] Full DJI M100 manufacturer specifications: https://www.dji.com/matrice100/info#specs

The Nvidia Jetson TX2[7] is a credit card-sized System on Chip (SoC) (50×87mm) designed specifically with providing dedicated on-board processing power in mind. What makes the TX2 special is its GPGPU compute capabilities, featuring 256 CUDA cores under Nvidia's new Pascal^{TM} architecture – rendering it suitable for the deep learning inference pipeline requirements of this project. Also an important consideration for mobile platform implementations (e.g. wheeled robots, UAVs) is its low power consumption, even under heavy computational load (maximum 15W[8]). The TX2 ships pre-installed on a fully-expanded development kit board manufactured by Nvidia, as illustrated in Figure 7.3a. As it would be impractical in both space and weight to install the full board onto the M100, instead a suitable carrier board – featuring fewer interfacing expansions – is used. The Connect Tech Inc. Orbitty carrier board[9] is specifically compatible with the Jetson TX1 and TX2 chips. It was selected for its minimal addition to the overall height profile (see Figure 7.3b) and range of required interfaces; Ethernet (for network communications with the DJI Manifold), USB (for serial communication with the M100's N1 flight controller), HDMI and SD-card (for recording flight data during experiments).



(a) Full Nvidia Jetson TX2 development kit.[10] *(b) Jetson TX2 installed on the Orbitty carrier board.*[11]

Figure 7.3: ***Selected UAV Computer.*** *The Nvidia Jetson TX2, selected On-board Embedded System (OES) for performing visual processing on-board the DJI M100 flight platform.*

Similarly, pictured in Figure 7.4, the DJI Manifold[12] is a small computer specifically designed to be installed on-board the M100 UAV. At its core, the Manifold is essentially a DJI-adapted version of the Nvidia Jetson TK1, the first generation of the Jetson series. Whilst it is not ideal to also house this computer on-board the M100 (given the additional weight, space and power consumption), the installation of this additional on-board computer is necessary for two reasons:

1. To access and decode the image feed provided by the X3 camera (see Section 7.2.1), a proprietary library operating on a specific decoding chip on-board the Manifold is the only way of doing so. Thus, the DJI Manifold computer is the only device capable of decoding the raw X3 image feed.
2. Computational load under full operation is intensive on a single device, especially resource-constrained computers such as the TX2. The addition of the Manifold permits some computation to reside on-board, somewhat alleviating the computational pressure.

[7] Nvidia Jetson TX2: https://developer.nvidia.com/embedded/develop/hardware
[8] Jetson TX2 manufacturer specifications:
https://devblogs.nvidia.com/jetson-tx2-delivers-twice-intelligence-edge/
[9] Jetson TX2 carrier board: http://connecttech.com/product/orbitty-carrier-for-nvidia-jetson-tx2-tx1/
[11] Image credit:
https://devblogs.nvidia.com/wp-content/uploads/2017/03/JTX2_Devkit-e1488775199359-624x615.png
[11] Image credit:
http://connecttech.com/wp-content/uploads/images/product-images/ASG003/ASG003_Orbitty-AngleA.jpg
[12] DJI Manifold: https://www.dji.com/manifold

Figure 7.4: ***DJI Manifold.*** *The DJI Manifold OES installed on-board the DJI Matrice 100 UAV flight platform. The computer is necessary to decode raw imagery received from the DJI-manufactured camera and gimbal system.*

Camera & Gimbal System

As an integrated camera and gimbal setup, the clear choice to accompany the M100 flight platform is the DJI Zenmuse X3[13]. The X3 is an integrated camera/gimbal system that captures high resolution video or images (camera pointed out in Figure 7.2a). Use of a gimbal allows the camera to be independent of the movement of the flight platform resulting in stabilised footage and reducing potential blurring. Equally, via use of the M100 API, one can issue absolute and relative rotational commands (for ϕ:roll, θ:pitch, ψ:yaw) such that the camera can be programatically pointed at objects of interest. Note also that the camera setup is physically attached to the flight platform via four rubber grommets to isolate in-flight motor vibrations affecting resulting imagery. Full specifications of the camera are given as follows in Table 7.2.

Parameter	Value
Diagonal Field of View	94°
Aspect Ratio	16 : 9
Maximum Resolution	4096×2160
Maximum Acquirable Resolution (via ROS)	960×720
ISO Range	$100 - 3200$
Roll Range	$\pm 90°$
Pitch Range	$[-90, 40]°$
Yaw Range	$\pm 320°$
Maximum Angular Velocity	$10\ °/s$

Table 7.2: ***Camera and Gimbal Specifications.*** *Parameter and value specifications for (top): the DJI Zenmuse X3 camera and (bottom): supporting 3-axis gimbal*[14].

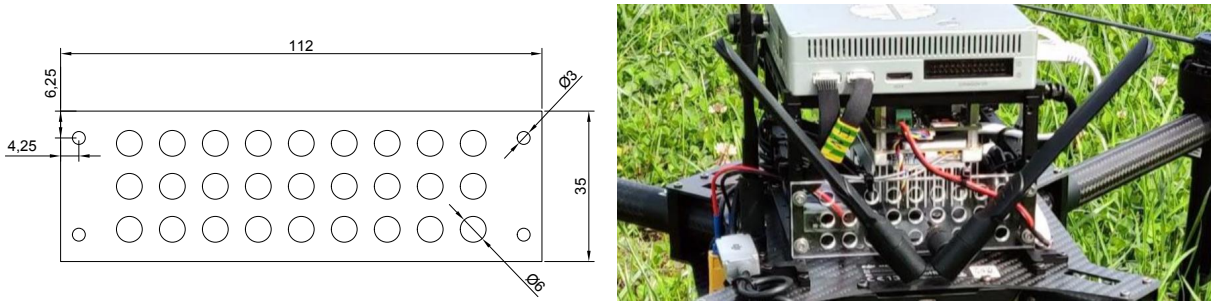
Modifications

Some modifications were needed in order to install and house the additional components on-board the flight platform described as follows:

- ***WiFi mounting plate***: with respect to the Jetson TX2, a custom mounting plate was designed and manufactured for its WiFi antennas permitting Wireless Local Area Network (W-LAN) communication with the basestation router and laptop (as shown in Figure 7.1). The plate is intended to

[13] DJI Zenmuse X3 integrated camera and gimbal system: https://www.dji.com/zenmuse-x3
[14] Full manufacturers specifications: https://www.dji.com/zenmuse-x3/info#specs

reside at the rear of the aircraft and conform to the dimensions imposed by the existing component racks. Design considerations included minimising overall weight whilst allowing maximum customisation with respect to antenna placement. Measurements were taken using electronic callipers and used to design the plate in Computer-Aided Design (CAD) software[15] shown in Figure 7.5a. The 2D design was then laser cut to clear 3 mm acrylic and mounted to the aircraft with WiFi antennas pre-installed (see Figure 7.5b).



(a) CAD drawing. *(b) Mounting plate in situ with WiFi antennas installed.*

Figure 7.5: ***Custom-made Antenna Mount.*** *(a): CAD drawing illustration and (b): image of the custom designed and built mounting plate for housing the on-board computer WiFi antennas.*

- ***TX2 power source***: a small additional device is needed in order to power the Jetson TX2 from the UAV's battery. This is necessary because the output voltage of the on-board Lithium Polymer (LiPo) battery (powering all components of the flight platform) is 22.2V, whilst the acceptable input voltage range of the Orbitty TX1/TX2 carrier board is $[9, 14]$V. To solve this problem, an intermediate voltage regulation device[16] is used to step down the voltage received from the M100 main battery to 12V. As part of this, a XT60 adapter was soldered to the connection with the battery via the M100's circuit board *and* an appropriately-sized male barrel connector to the power output for the TX2.

7.3 Dataset

A challenging effect of performing experimentation in a real-world setting is the lack of and difficulty in acquiring and labelling suitable training data for models involved in the experiment's full operational pipeline. Whilst for the case presented in Chapter 5, respective model training data can be synthesised easily alongside perfect ground truth labels, this is in reality obviously no longer the case. To this end – and much like the acquisition of the earlier AerialCattle2017 dataset (see Section 4.2) – several UAV flights were performed over a two-week experimental period consisting of two phases: *(a)* data acquisition and subsequent *(b)* conduction of experiments. In-between these phases, manual ground truth annotation took place alongside augmentations for the respective components. These flights were conducted in the same agricultural fields throughout this period on the same herd population of 17 yearling Heifer[17] Holstein Friesians. Comprising herd members are illustrated as follows in Figure 7.6.

Note that for all operations involving the UAV in flight above real cattle, appropriate ethical approval was sought and confirmed by the University of Bristol's animal ethics committee board. The purpose being to validate operational risks imposed by the UAV agent itself and its autonomous operation upon the welfare of the animals, and the measures put in place to satisfactorily alleviate these risks to a manageable level. Every flight was conducted in accordance with these constraints. The corresponding University Investigation Number (UIN) is: UB/18/064.

[15] Mounting plate designed in Draftsight: https://www.3ds.com/products-services/draftsight-cad-software/
[16] The Quanum QM12V5A-UBEC.
[17] 'Heifer' Oxford Dictionary definition: "a cow that has not borne a calf, or has borne only one calf".

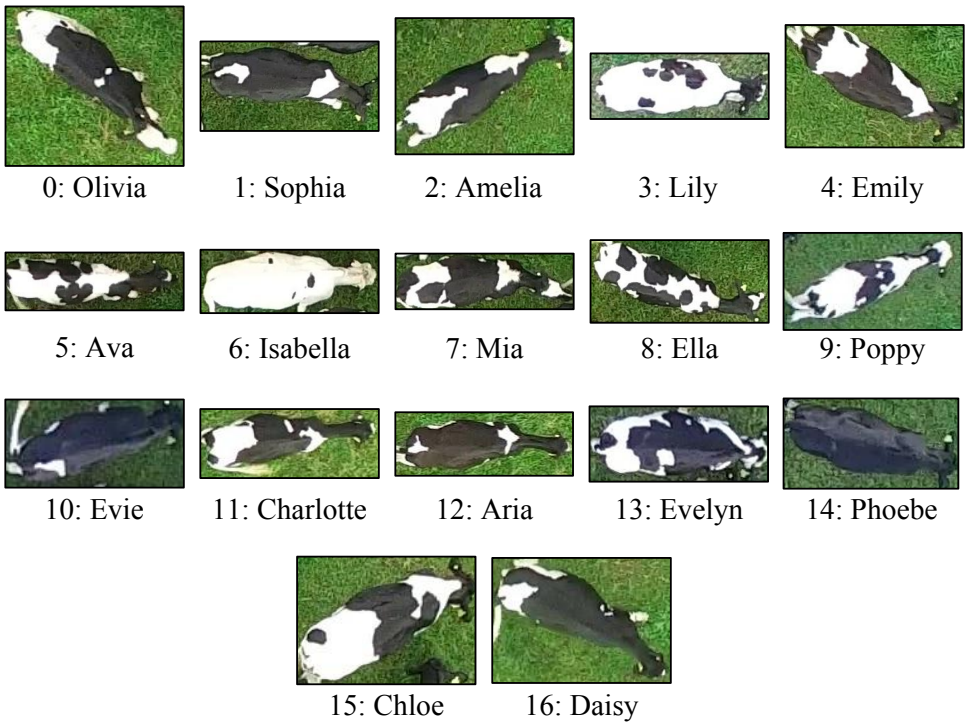


Figure 7.6: ***Final 17-Strong Cattle Population.*** *Sample still images of "ID: reference name" of all* 17 *yearling heifer Holstein Friesian individuals comprising the full herd population used for experimentation throughout this chapter. All* 17 *individuals are females that will be later used for milk production at the Wyndhurst Farm. This population was constant across the two week-long experimental period.*

7.3.1 Acquisition

Dataset acquisition was accomplished over two day-long recording sessions at the University of Bristol's Wyndhurst Farm in Langford Village, UK[18]. Fully manual and semi-autonomous flights were carried out over the acquisition period all whilst recording video at a resolution of 3840×2160 (4K) at 30fps via the on-board Zenmuse X3 camera gimbal system. The result was a raw dataset consisting of 37mins from 15 videos over 14 flights occupying 18GB (further statistics given in Table 7.3). The specific area of operation for UAV flights depicted in Figure 7.18 was kept constant during acquisition and throughout experimental testing also. Also importantly, the population of cattle consisting of 17 individuals was kept constant during this period too.

Motivated by previous successes in gradual acclimatisation of the animals to the physical and sonic presence of a large object in flight directly above them at relatively low altitudes ($\sim$10m) achieved for the AerialCattle2017 dataset (see Section 4.2), a similar approach was employed here. To remind the reader, this process involved taking off from a distant position in an adjacent field to a high altitude of 50m and gradually getting closer to the herd in three dimensions over the course of several flights. Video was captured during these acclimatisation flights yielding individuals to be resolved at low resolutions under a variety of viewpoints and poses. As behavioural assessments indicated a build up of comfort towards the UAV, increasingly high resolution imagery of individuals was acquired as a result of UAV-herd flight proximity. Since the algorithmic operation of identification components here intend to operate on completely aerial (straight down) viewpoints per individual, efforts were spent in animal acclimatisation on this class of flight behaviour. It was found that upon each subsequent visit, a short amount of time ($\sim$10 minutes) was required to reacclimatise the animals for that day session.

[18]Many thanks again to Wyndhurst Farm Manager, Kate Robinson and all others involved for facilitating UAV flights over the herd during the data acquisition and experimental period.

Figure 7.7: ***Example Aerially Acquired Frames.*** *Extracted frames from video acquisition flights illustrating the magnitude of variation of imagery owing to changing UAV height above the ground, illumination variation (e.g. casting shadows, low light) across the recording sessions as well as background change. Also visible is the significant challenge in both labelling and actual online model performance presented by the close proximity of individuals to one another. Each frame has annotated values ϕ, λ, h, θ approximating latitude, longitude, height above the ground and heading (where $N: 0, 360°, E: 90°, S: 180°, W: 270°$), respectively.*

7.3.2 Ground-truth Labelling

Given the acquired set of 15 raw videos, the ground-truth labelling process involves extracting and annotating appropriate data for the respective tasks and components of *(1)*: target detection yielding cow RoIs that are *(2)*: differentiated by the identity estimation model. This extraction and labelling process is visually summarised in Figure 7.9 and is described as follows with associated raw dataset statistics given in Table 7.3:

- ***(a)*** ***Frame Extraction***: constituent frames for each source video are extracted and saved to file at a rate of $1Hz$. This value is chosen to minimise subsequent manual labelling and annotation time whilst permitting sufficient time to elapse for image variation to occur (e.g. viewpoint change, animal movement, illumination variation). Extracted frames are manually inspected to assure the presence of labelling material (some cow(s) are present in the image), frames that do not fit this criteria are discarded.
- ***(b)*** ***Bounding Box Labelling***: every resulting frame (containing at least one target) then has ground truth bounding boxes manually annotated via a custom GUI (as for the labelling process for earlier datasets in this thesis; see Section 3.3.2).
- ***(c)*** ***Region Labelling***: At this point, the now rectangularly annotated frames are still at the high source resolution of 3840×2160 pixels. However, images acquired online/on-board the UAV are maximally captured at 960×720 pixels. Since flights will be performed at $\sim$10m in height above the ground, cattle targets are resolved at $\sim 150 \times 150$ pixels (under the assumption that cows are approximated to be $\sim 2 \times 2$m) – more on this in Section 7.4. Thus, it is desirable to provide training data in which cattle are resolved to $\sim 150 \times 150$ such that the model learns features at the same resolution inference will be performed on. Correspondingly, and, since input images are non-proportionally resized to fixed input tensor size for the YOLO detector (set to 736×736 to satisfy divisibility by 32), square regions are extracted via another custom GUI (Figure 7.8 illustrates the annotation interface). In doing so – given that data acquisition flights were performed at higher $\sim$20m height above the ground for animal acclimatisation to the UAV – targets were approximately resolved at the correct dimensions.
- ***(d)*** ***Manual Individual Identification***: alternatively to region labelling for target detection, target centric RoIs are manually identified here for input as training data into the identity estimation model.

Figure 7.8: ***Region Labelling GUI.*** *Example screenshot of the custom-built GUI for manually annotating* 736×736 *pixel square regions for input as training data into the cattle detection model. Annotated ground-truth target labels are show in red and the user is asked to click and drag square regions (shown in green) over targets and accept them (blue regions) until all targets have been covered. Each accepted image region is saved to file alongside region-relative transformed target bounding box annotations. Note that targets are sometimes naturally duplicated across saved regions, whilst partially occluded annotations are discarded altogether.*

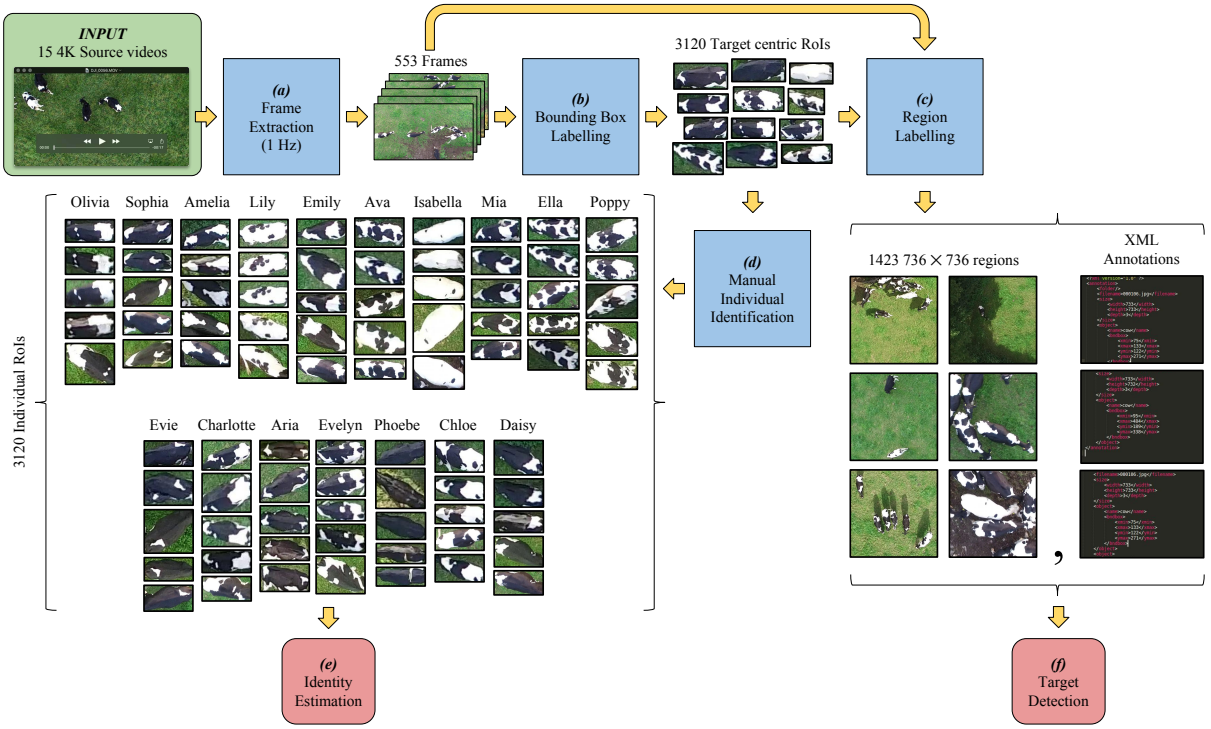


Figure 7.9: ***Full Dataset labelling process.*** *Illustration of the complete ground truth dataset labelling process for the identity estimation and target detection models from source input* 3840×2160 *resolution videos acquired using the UAV. At each stage, the number of frames/regions/RoIs is listed as according to Table 7.3.* ***(a)****: individual frames are extracted from source videos at* $1Hz$ *with solely background frames (containing no cattle) manually discarded. Next,* ***(b)****: target bounding boxes are manually annotated yielding a set of target-centric RoIs that are either* ***(d)****: manually identified for input as training data into the* ***(e)****: identity estimation model or* ***(c)*** 736×736 *pixel regions are manually annotated such that targets are resolved at a desired resolution of* $\sim 150 \times 150$ *pixels (given a UAV flight altitude of* $\sim 20m$*) for training input into* ***(f)****: the target detection and localisation model.*

	Videos	Extracted frames	Usable frames	Extracted Square Regions	Individual RoIs
Number	15	2285	553	1423	**3120**
Storage space (GB)	18.3	9.5	2.4	0.4	0.03
Time (mins)	37	-	-	-	-

Table 7.3: ***Dataset Labelling Statistics.*** *Table of various statistics regarding the dataset size at specific points in the extraction and labelling pipeline. Overall, 37 minutes of video were considered for labelling, resulting in* 1423 *target detection examples and* 3120 *individual-centric RoIs. Extracted square regions have dimensions* 736×736 *to match the acquirable image size from the UAV's on-board camera.*

7.3.3 Augmentation

To synthesise additional data yielding a larger data corpus for respective ANN model training – alleviating model over-fitting and improving model generalisability and robustness – image augmentations are performed on original imagery[19] over both: target detection *and* identity estimation datasets resulting from the aforementioned labelling pipeline. With respect to identification data, the purpose of augmentation here also being to balance the number of training instances across the population. The per-class distribution of original non-synthetic images is shown in Figure 7.10. Example augmentations are shown in Figures 7.11 and 7.12 towards the task of target detection and individual identification, respectively.

[19]Image augmentations are implemented by the 'imgaug' project published at: https://github.com/aleju/imgaug

Augmentations for both detection and identification training datasets are performed stochastically with the possibility for any combination of the operations listed as follows according to a per-operation likelihood value: horizontal & vertical flipping, crop & pad, affine transformations (scale, translate, rotate, shear), Gaussian, average or median blurring, noise addition, contrast variation and small perspective transformation. Note that ground truth bounding box annotations are also transformed according to the random augmentation sequence.

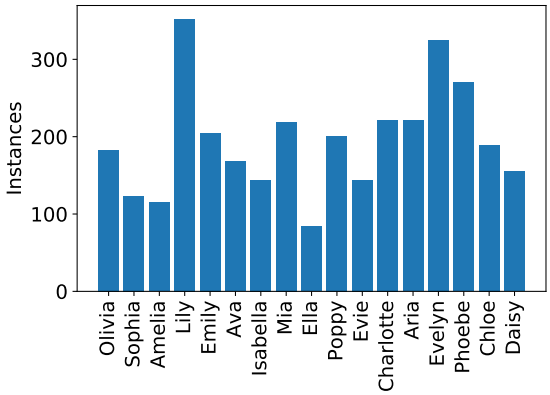


Figure 7.10: ***Original Instance Distribution.*** *Distribution of real, non-synthetic images across each of* 17 *possible cow identities with mean* $\mu = 195.3, \sigma = 68.99$ *and median*= 189. *Note that this graph indicates which individuals exhibited higher comfort towards the presence of the UAV, as reaffirmed later in Section 7.6.*

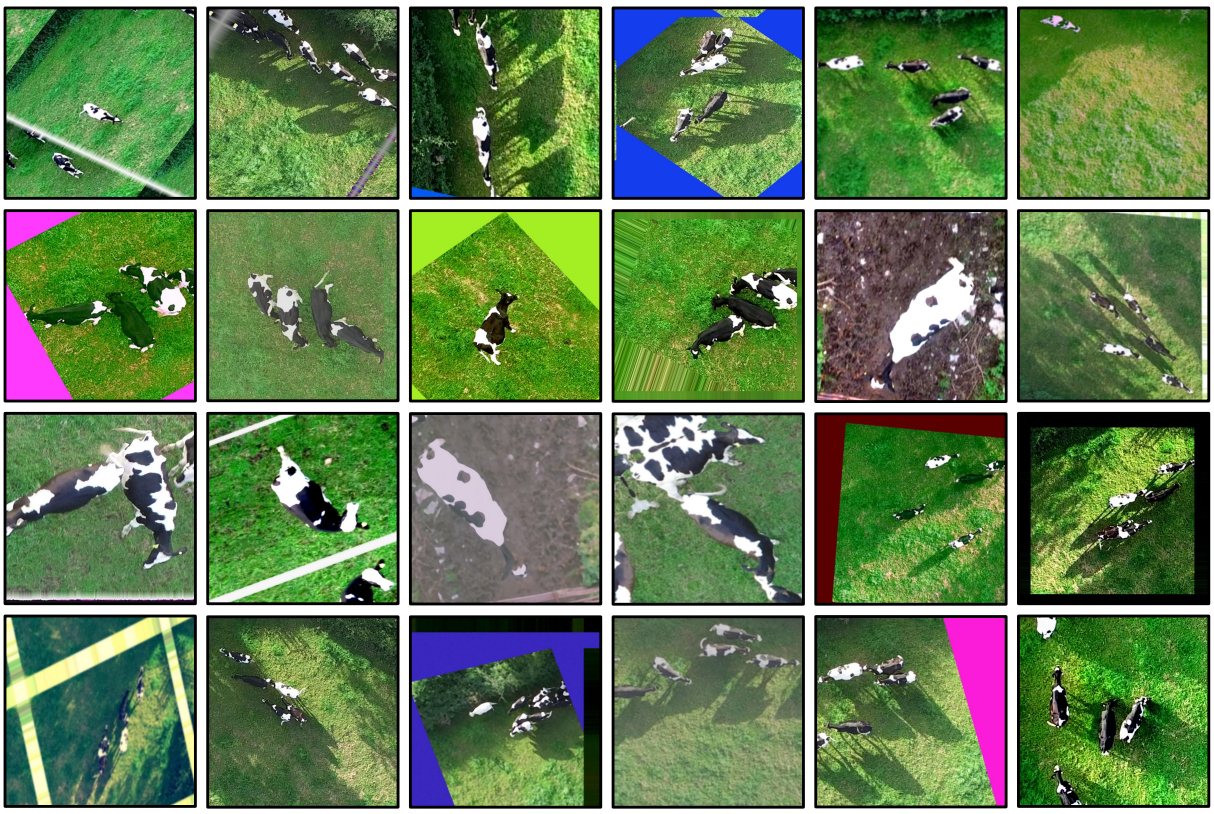


Figure 7.11: ***Target Detection Augmentations.*** *Examples of image augmentations for* 736×736 *pixel regions that – in conjunction with augmented ground truth bounding box annotations – are provided as training instances for the detection model founded upon the state-of-the-art YOLO detector [257].*

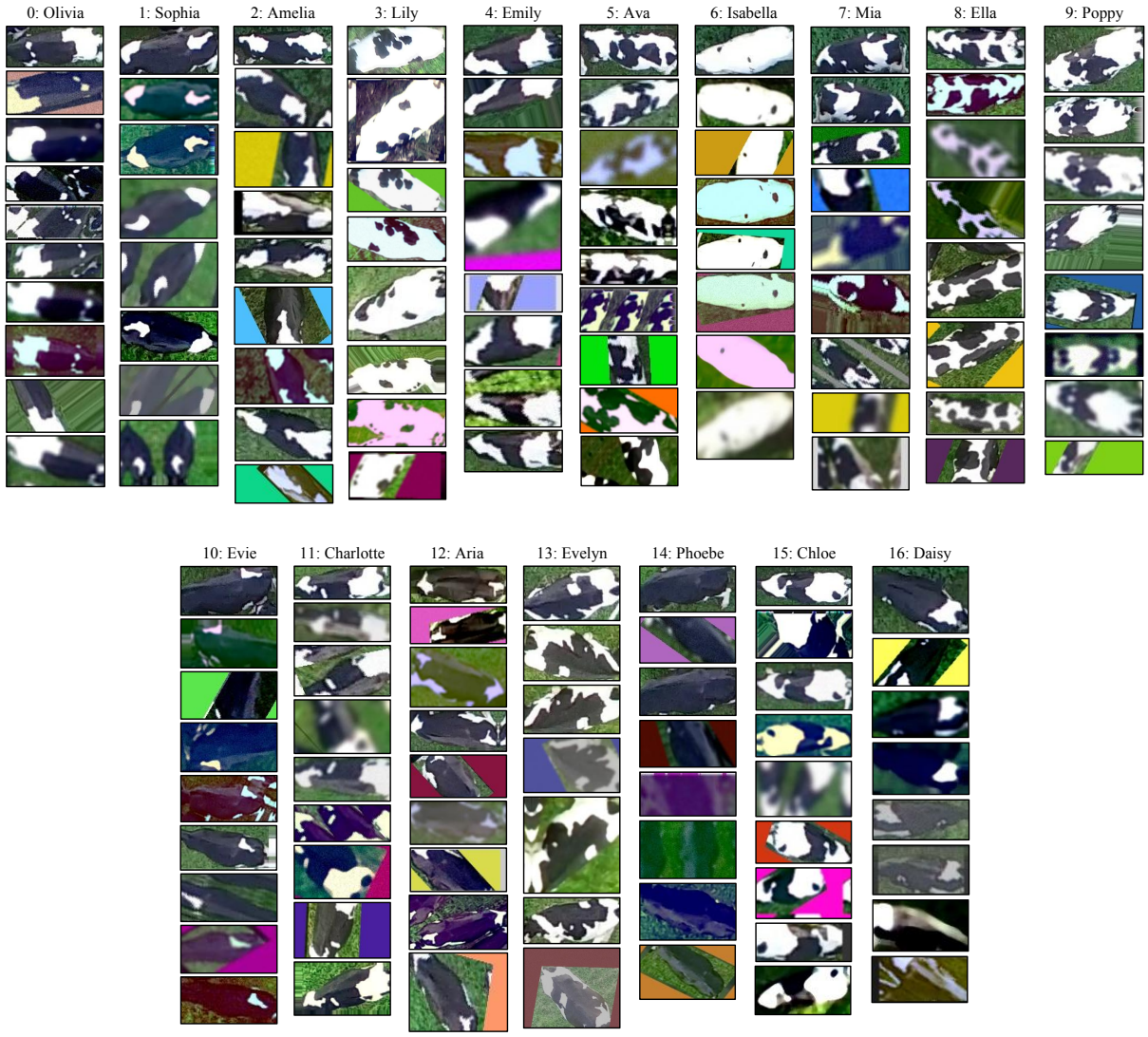


Figure 7.12: ***Individual Augmentations.*** *Random examples of image augmentations on the final training set comprised of individual-wise RoIs. For each identity, the top image is a random non synthetic example from which augmentations will have been initialised on.*

7.4 Model Implementations

In this section, implementation details are given for the components, pipelines and models that together comprise the software that operates on-board the UAV flight platform for real, online experiments performed in this chapter. This mostly consists, here at least, in giving details on the modifications made to transition from operation on simulated inputs to a real-world agricultural environment and limited computational resources. An overview of the complete online experimental pipeline operating on-board the UAV is illustrated as follows in Figure 7.13. Principally, the goal is to perform exploratory agency in reality whilst passively and iteratively identifying targets as they are discovered within the environment. At each exploratory iteration (fulfilment of an exploratory action e.g. up, down, left or right one grid cell), $n = 5$ samples are taken over a short period of time to allow (a): observation variation of targets to occur yielding improved identification success and (b): a better global position estimate via averaging multiple GPS samples over time during which the UAV's position will settle in each dimension (e.g. recovering from possible PID controller overshoot).

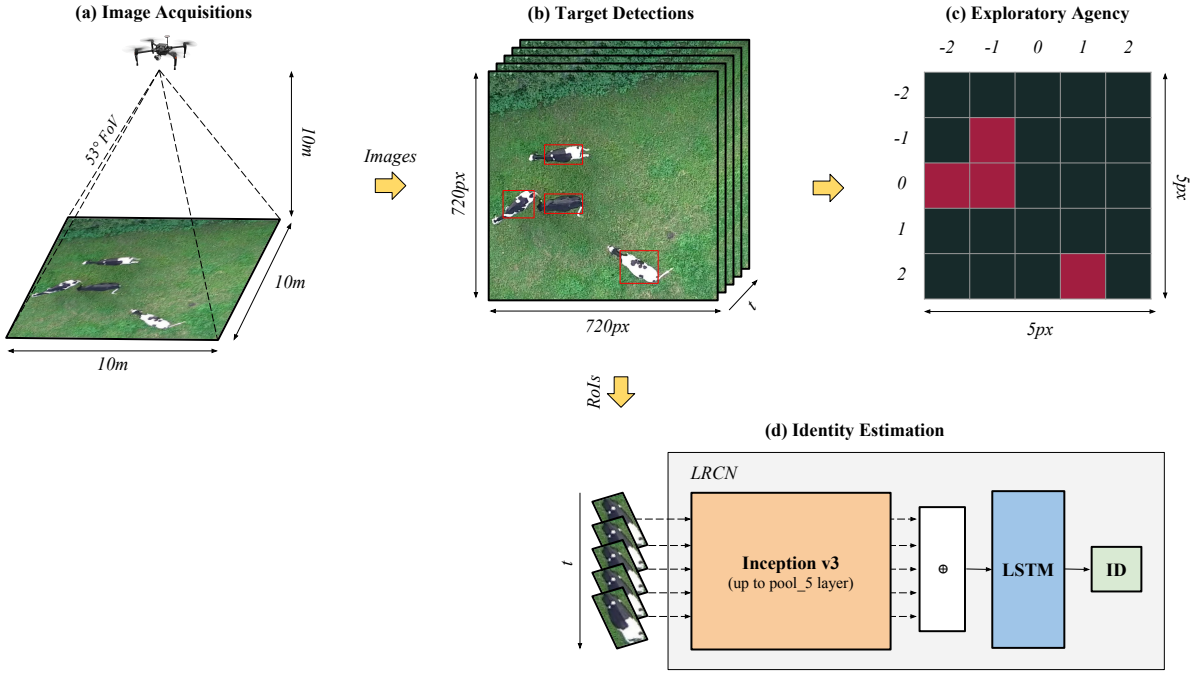


Figure 7.13: ***Full Experiment Pipeline.*** *Illustration of the complete experiment pipeline*[20]*. Firstly (a); image acquisition takes place, whereby $n = 5$ image and flight platform position samples are taken over a short period of time and (b): targets are detected on each individual image (Section 7.4.1). Resulting target-centric RoIs are then either (c): temporally filtered for the best (most confident) detection per grid position to yield the 5×5 pixel image given as input to exploratory agency (Section 7.4.2), or (d): spatially consistent RoIs over time are concatenated to form input into the identity estimation pipeline utilising a LRCN (Section 7.4.3).*

7.4.1 Target Detection and Localisation (TD)

As for earlier chapter (see Section 5.4.1), target detection and localisation is performed by state-of-the-art detector YOLO v2 [257], as opposed to solutions founded in Faster R-CNN employed in earlier chapters (Section 4.3). The motivation being similar or better qualitative and quantitative performance [257], coupled with faster, even real-time inference – especially important for online computation in this chapter. The model is retrained here from scratch on the aforementioned dataset consisting of $11,384$ synthetic and non-synthetic training images and associated ground truth labels. As a result of expecting 720×720 images from the UAV camera, the input tensor size was modified to 736×736 (sizes must be a multiple of 32) in order to match individual resolution in training and testing at the slight cost of increased inference time from additional parameters.

Model inference on an image I yields a set of m bounding boxes $B = \{b_0, b_1, ..., b_m\}$ with associated object confidence scores. Inference on each of $n = 5$ image samples then produces this set over time $\{B_0, B_1, ..., B_n\}$. Since all samples are taken over a short period of time (e.g. $\sim [1,2]$s) when the agent is hovering, targets present in I_j should also be present in I_{j+1} under some viewpoint variation that is beneficial for identification. Association of detections over time (object tracking) is achieved via a simple spatial model that splits the image equally to match the input size of the exploratory agency algorithm visual input abstraction – set to 5×5 pixels in the case here, more on this in Section 7.4.2. If identification estimation is requested (as indicated by exploratory agency), the RoIs corresponding to the most confident central cell detections for each image $\{I_0, I_1, ..., I_n\}$ are concatenated to be temporally consistent for input into the identity estimation model. With respect to formulating the 5×5 pixel visual

[20] UAV image credit: DJI

sensory abstraction for exploratory agency input, the most confident object RoI of each of 25 144×144 pixel cells over the n images are given to binarily indicate target presence. Whilst the MOT model utilised here is relatively primitive, it is important to re-iterate that the goal of this chapter is to provide a proof-of-concept and that computational resources are limited on-board the UAV. Investigations into the viability of replacing the employed strategy with more complex algorithms operating in the predominant "tracking by (re-)detection" paradigm [278, 29, 41, 51] is certainly a possible avenue of future work.

7.4.2 Exploratory Agency (EA)

Modifications in transitioning the exploratory agency architecture and paradigm described in Chapter 6 to the real-world scenario are relatively minimal. To remind the reader, the employed state-to-action architecture consists of a dual-stream CNN with the two following inputs (for more specific details, see the original Section 6.3):

1. ***Image*** I: visual sensing abstractions local to (centred about) the agent. In the case here, this is a $x \times y$ pixel image generated from target detections in the original, UAV-acquired image – as for the non-simulated scenarios in Chapter 6 (see Figure 6.3 for an example). This generation process consists of centrally cropping the 960×720 image obtained from the M100 flight platform and the Zenmuse X3 camera/gimbal system down to square resolution 720×720 (since exploratory grid cells are also square) and then linearly scaling to 736×736 for direct input into target detector YOLO. The image is positionally split into x equally-sized columns and y equally-sized rows. Target detections indicated by YOLO situated in particular cells – considering bounding box centre-points – form coloured pixels in the resulting $x \times y$ matrix given as input to the exploratory agency network. An example of this process is given in Figures 7.13 and below in 7.14. In this simplified form, the exploratory model can be trained easily in simulation and generation of appropriate training data is not constrained by the difficulty in acquiring and labelling sufficient real imagery of cow targets.

Figure 7.14: ***Exploratory Agency Image Input Generation Process.*** *Pipeline for generating images ultimately provided as one of the inputs into the exploratory agency model. The image is split into* 5×5 *equally sized cells, detections that fall into a particular cell are correspondingly marked in a new* 5×5 *binary pixel image forming input* I *into the EA model. Small pixel images in this form are identical to the sensory abstractions utilised throughout Chapter 6.*

2. ***Occupany map*** M: an evolving spatial map of the complete environment encoding the current agent position, past visited locations and discovered target positions. To enable this, a two-dimensional rectangular grid must be fitted to the experimental flight area such that the dimensions of the exploratory map M can be defined spatially by parameters $w \times h$. This fitting process is obviously experimental area-specific and with respect to the field in which experiments take place, is described generally in the following Section 7.5 and specifically in Section 7.5.2. The determination of these parameters then directly define the dimensionality of input M into the secondary CNN that examines current and past positions to inform corresponding agent navigational decisions (see Figure 6.1 for the full dual-stream architecture). Note that the dynamic, spatio-temporal

occupancy map extension used in previous Chapter 6 is not used here to maintain model operational simplicity. Instead, when some $\delta\%$ of the $w \times h$ map has been explored, it is simply reset altogether. This behaviour is implemented since targets may now have moved and thus, previously explored map regions may now contain target(s) one wants to discover.

Note that throughout experimentation in simulation conducted for Chapter 6, the agent's initial position G_x, G_y within the exploration grid was always randomised to yield model generalisability. This is now no longer the case here; the UAV agent takes off and lands from a defined location and proceeds to another which is some edge coordinate of the exploratory grid. To model this behaviour accurately therefore, in the synthesis of training data for exploratory agency, the agent's initial position is created equal to that in reality (respective to the experiment location, see Section 7.5.2).

7.4.3 Identity Estimation (IE)

Identity estimation is performed by the LRCN architecture here, identically to previous Chapter 5, with substitution of simulated individuals for the aforementioned augmented dataset of 17 real individuals (see Section 7.3). Accordingly, the procedure for training is also equivalent – refer to Section 5.4.4 for a full description of the training algorithm. To summarise; (1): a GoogLeNet/Inception CNN [302] is trained on individual RoIs non-proportionally resized to 224×224 pixels with 17 possible classes/identities, (2): $n = 5$ same class randomly selected RoIs from the same respective categories are passed through the trained GoogLeNet until the last layer and feature vectors are combined over the n samples. Finally, (3): a shallow LSTM network is trained on these sequences of feature vectors.

Figure 7.15 provides evidence of per-category learning of appropriate spatial representations using local interpretable model-agnostic explanations [260] – qualitatively highlighting the success of the Inception architecture learning discriminative and fine-grained visual features for each individual. Figure 7.16 illustrates learnt convolutional filters, whilst Figure 7.17 presents quantitative findings on training and validation accuracy versus epochs for CNN and later LSTM training, respectively. As can be seen, GoogLeNet trained to identify individuals from single instances achieves 97.13% accuracy on the validation set whilst the LSTM subsequently achieved 100% after just the first epoch. These results suggest that in the vast majority of cases, a single iteration evaluation approach is successful, whilst evaluation of multiple variant iterations permits the final accuracy barrier to be overcome.

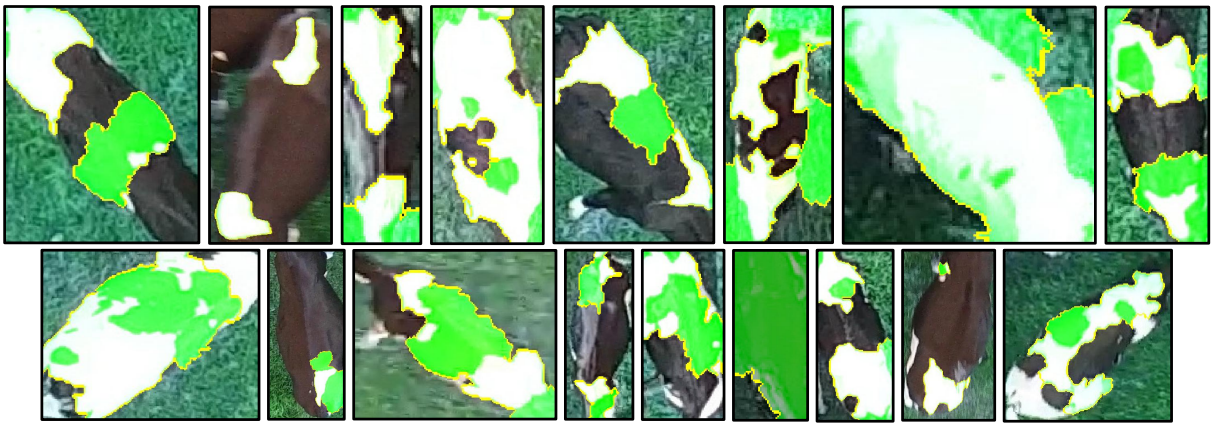
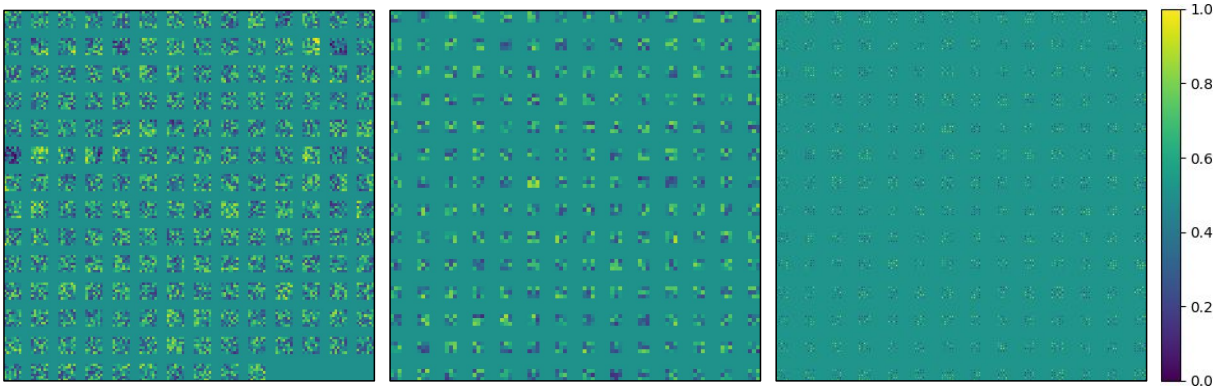


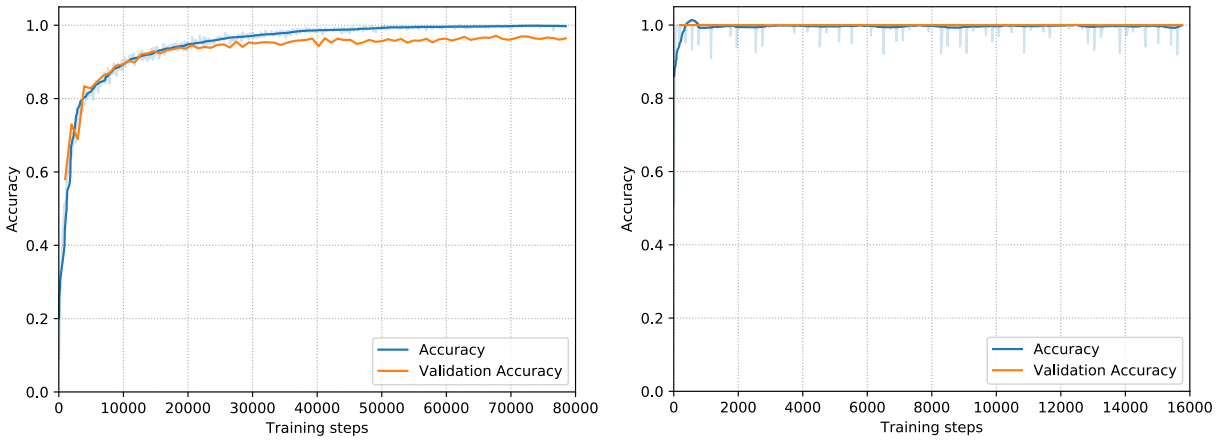
Figure 7.15: ***Per-Class Local Interpretable Model-Agnostic Explanations.*** *Hand-picked local interpretable model-agnostic explanations [260][21] for each possible cow identity/class/category illustrating the success of CNN training in learning per-category discriminative features. Green regions for each image depict the superpixel(s) that were activated in order to yield correct predictions from the Inception/GoogLeNet CNN (not the later trained LSTM) trained on the* 17*-strong population.*

[21] Images were created using the released LIME implementation: `https://github.com/marcotcr/lime`.



(a) Conv1: 7×7 *(b) Conv2: 3×3* *(c) Inception 4e 11*

Figure 7.16: ***Learned Convolutional Filters.*** *Examples of high, mid and low levels of learned convolutional filters for the identity estimation network architecture – based on GoogLeNet [302] – trained on cow-centric RoIs from the 17-strong herd population. Filters have been normalised into visible range* $\in [0, 1]$.



(a) GoogLeNet CNN accuracy *(b) LSTM accuracy*

Figure 7.17: ***Identity Estimator Training Graphs.*** *Training and validation accuracies versus training steps for the feature extraction CNN with GoogLeNet/Inception [302] architecture providing concatenated feature vectors from inference on 5 randomly selected same-class* 224×224 *RoIs to the LSTM for iterative identity recovery. In this combination, the pipeline forms a LRCN [73] architecture. Training signals have been smoothed using the Savitzky-Golay filter [275] for visualisation purposes.*

7.5 Experimental Setup

7.5.1 Geofence

A geofence is a virtual geographic boundary defined by a set of GPS coordinates consisting of latitude, longitude and altitude. The purpose of such a boundary within experiments conducted here is to provide ultimate supervisory control of the flight platform in case of failure, as is commonly implemented in UAV-based literature [122, 15, 349]. The intention is that the UAV never actually violates this boundary, but is there as a last resort in case of software errors, bugs or otherwise. If the boundary is positionally violated, an error is indicated and corrective procedures are enacted – typically halting the vehicle at that boundary violation; the chosen reactive measure here.

Its implementation here begins with the user manually defining the acceptable flight region via a set of GPS coordinates that together form a polygon for the respective experiment area – Langford and Long Ashton in the case here. Note that as can be seen in Figure 7.18, the drawn cordon includes the experiment area itself, the operation zone or 'runway' (for takeoff and landing) and the corresponding passageway between them. This is achieved via the use of freely available software Google Earth Pro, where the user can manually annotate and label perimeter coordinates and export these in a common shape and Geographical Information System (GIS) file format. Then, at a rate of $1Hz$, the UAV's current GPS coordinates are queried to determine whether they reside inside or outside of the geofence boundaries via API calls to GeoPandas[22]. If the UAV is determined to be within bounds, the experiment continues as normal, if not the user is indicated an error and flight halts immediately to hover at the violation position. The intention is that the human operator then manually controls the flight platform back to its landing zone so that the cause of the problem can be investigated via inspection of flight logs and otherwise.

Note that this GPS-based geofence only affects the horizontal xy (east-north) plane and that similar software errors/bugs may cause boundary violations in the vertical dimension z or 'up'. Whilst a 3-dimensional geofence could be created, the DJI developer API implements a function to receive the current height above the takeoff point. This value is also inspected at $1Hz$ when the UAV is situated within the experiment area to ensure it does not violate a lower bound posing a threat/risk to cattle targets whilst an upper bound is verified at all times.

Figure 7.18: ***Langford Enforced UAV Flight Geofence.*** *Illustration[23] of the enforced polygonal geofence (transparent white area) defined by a set of GPS coordinates in Google Earth for the actual experiment location at Wyndhurst Farm, Langford Village, UK. If the UAV violates these boundaries, flight halts to hover at the violation position and control is handed back to the supervising human operator. The employed geofence is selected here to comply with CAA flight distance regulations from elements outside of operational control [16] (e.g. at least* $50m$ *away from public roads, private houses) whilst simultaneously ensuring the flight platform does not collide with obstacles (high hedges, trees, etc.). Also illustrated are the two possible runways for takeoff and landing (aeroplane symbols), corresponding experiment start positions (white markers) and the fitted* $40 \times 40m$ *exploratory grid/map (indicating map scale).*

[22] GeoPandas: http://geopandas.org/

[23] Image courtesy of Google Earth Pro: https://www.google.com/earth/

7.5.2 Exploration Map Fitting

Further to the discussion in Section 7.4.2, it is necessary to fit a 2-dimensional grid to experiment and flight operation areas to allow exploratory agency to take place (refer to previous Chapter 6). The fitted grid forms the basis upon which the UAV-based agent explores the environment in question to discover the positions of targets of interest (i.e. Holstein Freisian cattle). Whilst a simple lawnmower pattern could be employed, as previously motivated, constraints imposed by the UAV's on-board battery capacity (especially under high computational load) justify the need for fast search methods. The grid is fitted with respect to the experiment area/field, and is illustrated above in Figure 7.18 and in Figure 7.19 for both the flight testing zone (see Section 7.5.5) and the actual experiment area on real cattle.

Similarly to the previous section describing geofence creation, per-location exploratory grids are established utilising Google Earth Pro. To begin, the user manually annotates the grid's origin in the form of a single GPS coordinate. Subsequently, a decision is made regarding the size of the square $s \times s$ m grid cell that the agent and targets are resolved to positionally with respect to the exploratory grid (G_x, G_y). From the created grid origin GPS coordinate, orthogonal lines are drawn of length $s * w$ and $s * h$ for the horizontal axes x and y, respectively, where the exploratory grid then contains $w \times h$ grid cells of size $s \times s$ metres. These components together form the exploratory grid that can then be resolved to four GPS coordinates defining the perimeter of the experiment area that the agent should not violate apart from fulfilling takeoff and landing. Should this not be the case, the geofence – defined to marginally encompass this experiment area – provides ultimate position error handling.

Another consideration is that local Cartesian position offset commands are fulfilled by the M100 flight platform in a reference frame aligned with ENU and that the fitted exploratory grid is unlikely to be aligned with this frame. As such, the heading of the grid's positive x-axis is determined[24] ψ_{EA} and is used to transform exploratory actions and otherwise performed by the agent into the ENU frame. That is, an agent-centric local position offset command $\vec{V} = [\Delta x, \Delta y, \Delta z]$ is transformed two-dimensionally via:

$$\begin{bmatrix} \Delta x' \\ \Delta y' \end{bmatrix} = \begin{bmatrix} cos\psi_{EA} & -sin\psi_{EA} \\ sin\psi_{EA} & cos\psi_{EA} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{7.1}$$

and

$$\Delta z' = \Delta z, \tag{7.2}$$

since the exploratory grid vertical or 'up' dimension remains aligned with ENU. Note that usually, this process should be performed with respect to the agent's current attitude (yaw, heading), however, in all experiments here the agent's yaw angle G_ψ is always kept aligned with the exploratory grid's x-axis and is thus equivalent.

	s(m)	w	h	$O_{long}(^\circ)$	$O_{lat}(^\circ)$	$\psi_{EA}(^\circ)$
Fenswood Farm, Long Ashton, UK.	2.5	10	10	51.234	−2.71289	−68
Wyndhurst Farm, Langford Village, UK.	2	20	20	51.34	−2.77161	−180

Table 7.4: ***Exploratory Grid Parameters.*** *Table of selected exploratory grid parameters for experiment areas: Fenswood and Wyndhurst farms conducting flight platform testing and real, live experimentation, respectively. Parameter definitions: s denotes the square length in metres of a single grid cell that targets are resolved to, w, h define the width and height of the fitted exploratory grid environment, respectively. Parameters (O_{long}, O_{lat}) together define the origin of the exploratory grid defined as a GPS coordinate with a corresponding orientation (ψ_{EA}) with respect to the 'up' axis. The important choice to note here is that the fitted exploratory domain consisting of $20 \times 20 = 400$ cells for real experiments at the Wyndhurst Farm is significantly larger than the $10 \times 10 = 100$ grid used at Fenswood here and throughout Chapter 6 in simulation.*

[24] Utilising Google Earth Pro.

(a) Fenswood farm, Long Ashton, UK. **10 × 10 *grid*.** *(b) Wyndhurst farm, Langford Village, UK.* **20 × 20 *grid*.**

Figure 7.19: ***Fitted Exploration Grids.*** *Illustration of the manually fitted exploratory grids for each flight experiment location according to the parameters defined in Table 7.4. In both images, true north is directly upwards with blue dots marking possible grid positions. White squares show the grid origin along with cell dimensions. Note that the images are not shown at the same scale and were produced via Google Earth Pro.*

7.5.3 GPS Coordinate Fulfilment

Throughout the operation of live flight experiments, the UAV-based agent is frequently issued commands to fulfil particular manually defined GPS coordinates (e.g. takeoff/landing zone, experiment start position). Frustratingly, at the time of writing, the DJI API currently does not feature a corresponding function call to realise a target coordinate in GPS. Instead, position commands are issued to the M100 flight platform via local position offsets in metres with respect to a programatically-set ENU reference frame. As such, in order to fulfil a target GPS coordinate, it must be converted into that same frame. This is achieved by converting the target GPS coordinate into the static ECEF reference frame, then converting that coordinate into the local ENU frame. Equally, the same process is performed on the agent's current GPS position and the resulting local positions are compared. The implementation of this process – following common standards established in literature [25, 137] – is fully described in Appendix B. Note also that throughout this thesis, all references to GPS coordinates refer to the WGS-84 GCS standard [63].

7.5.4 Flight Simulator

A key component of flight and experiment verification was the use of a comprehensive simulation environment. Accompanying the DJI M100 UAV is a piece of software that permits the user to set aircraft parameters necessary for flight (e.g. application key and ID, GPS module mounting position, baud rate for serial communication) along with a program to simulate UAV flight. One interacts with the flight platform in the exact same manner in reality; via the controller as expected, or using API calls to the onboard flight controller. Thus, software written in this chapter can be fully verified in simulation, which, especially when dealing with limited battery time, difficulty in suitable weather windows and potential code bugs crashing the UAV renders the simulation package invaluable during development. Figure 7.20 illustrates the UAV in simulated flight and the accompanying hardware setup. To establish this operational mode, a USB cable connecting the M100's N1 flight controller/autopilot and a host simulation

computer is established. Within the simulation software, the user can manually specify parameters such as the aircraft spawn GPS coordinates, per-dimension wind speeds, etc.

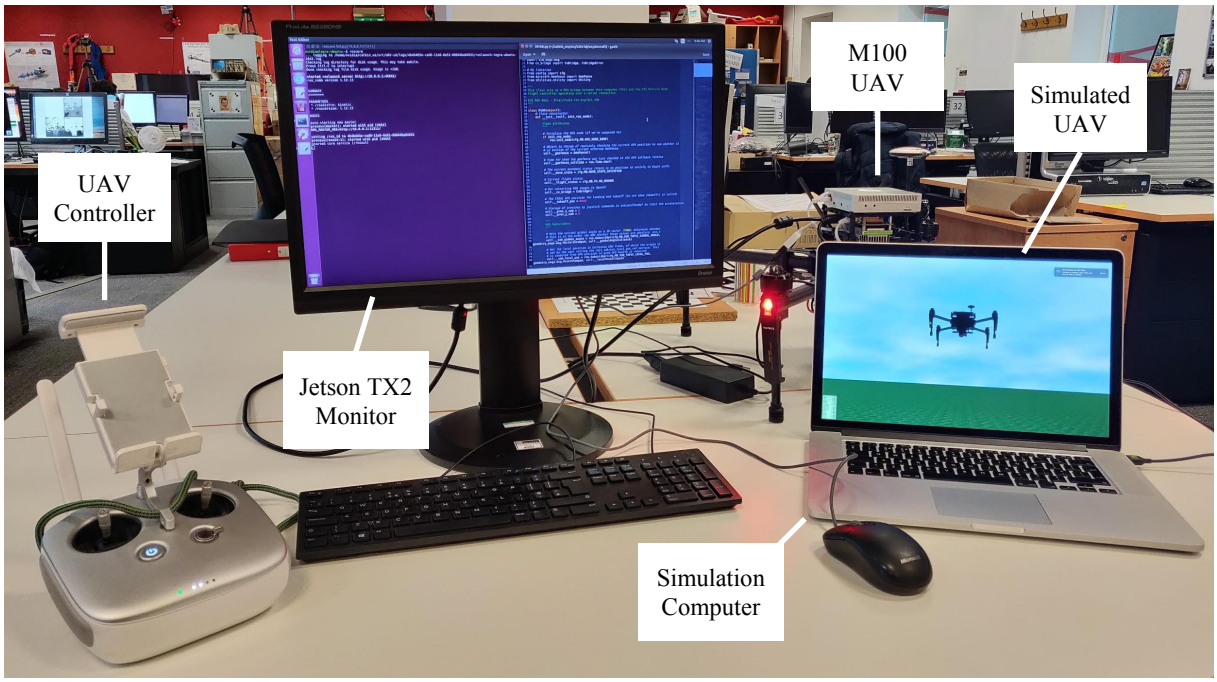


Figure 7.20: ***Flight Simulator Setup.*** *The comprehensive flight simulation environment setup permitting software interactions with the on-board flight controller via the DJI API to be debugged without testing in reality – every interaction is identical to the real case. The simulation computer running proprietary DJI software connects to the M100 via USB. The UAV can be flown entirely manually or via function calls through ROS to the DJI API, as is the case here for autonomously-operating flights.*

7.5.5 Flight Testing Area

As indicated earlier, initial testing flight experiments (without real cattle) were conducted in a separate location at Fenswood Farm in Long Ashton, UK. The purpose being to verify UAV flight components and automations without the added risk to cattle welfare in the event of failure. Additionally, initial experiments in a non-critical setting allow operation procedures, kit lists, error handling, etc. to be established and practised such as:

- ***Flight logs***: appropriate flight experimentation documentation is inherently important for accountability and data attribution (e.g. total flight time, battery number, unique experiment ID). To accomplish this goal, experiment details are annotated per-flight on a written sheet mounted on a clipboard.
- ***Data records***: per-flight, the on-board computer logs data relevant to the experiment itself (e.g. imagery, position fulfilment requests, internal exploratory map memory) to a SD card such that experimental performance can later be analysed and/or reproduced offline.
- ***Item checklist***: an item checklist was established to ensure all components are correctly brought for experiments (e.g. spare batteries, propellers, a small toolkit, remote controller).
- ***Hardware operation***: it was found that the order in which components are turned on, activated, shutdown, etc. is of crucial importance. Thus, orderings and dependencies were established in written form for future experiments.
- ***Geofence verification***: to verify geofence violation handling, a smaller testing geofence was created and the UAV was commanded to fulfil a GPS coordinate outside the geofence.

[25] Special thanks to Dr Matthew Harper for permitting use of his digital camera.

Figure 7.21: ***In-Flight Images.*** *Pictures of the M100 UAV in flight during initial testing[25] of fully manual (via the controller) and semi-manual control (via keyboard operation) at the University of Bristol's Fenswood Farm, Long Ashton, UK to establish and verify manual and autonomous flight control prior to operating experiments with increased risk due to the presence of live cattle.*

- ***Ultimate aircraft control***: whilst the intention is for completely autonomous flight to take place, it is important to understand how and to what extent the human operator can exercise ultimate aircraft control in case of critical failure, software bugs, etc.
- ***WiFi range***: as illustrated in Figure 7.1, a W-LAN connection between the operator's laptop and the on-board Jetson TX2 computer is required for experiment monitoring and more. To establish the maximal possible range between the UAV and the WiFi router whereby communication still occurs successfully, the UAV was carried directly away from the router. Meanwhile, the operation laptop sent ping commands to the TX2 at $1Hz$. It was found that commands would be reliably received (albeit with high latency) up until $\sim$150 m distance between the UAV and the router (with direct line of sight; no obstructions). The distance itself was measured via the DJI Go Android application comparing GPS coordinates.
- ***Battery capacity***: an important factor for conducting experiments is the capacity of the battery and consequently the total possible flight time of the flight platform under normal use. Over the flights that were performed here, a maximum flight time of $\sim$11 minutes was observed (including takeoff/landing) despite the heavy payload and high computational expense, which is in keeping with the manufacturer-suggested flight time of approximately 13 minutes with a 1 kg payload (see Table 7.1).
- ***Fail-safe mechanisms***: situations where some failure occurs are dealt with employed fail-safe mechanisms that were verified here in case of: loss of WiFi, critical battery charge (if $\leq 20\%$, action: return to home), and more.
- ***Reference frame verification***: differing reference frames for position commands, etc. are easily confused during development and thus, must be verified in reality.
- ***GPS coordinate realisation***: fulfilling requested GPS coordinates is more complex than at first appearance due to a lack of API functionality on-board the flight platform. Instead goal coordinates are transformed to local position offsets via the process described in Section 7.5.3 and Appendix B.
- ***Local position realisation***: verification of accurate local position realisation via the implemented control mechanisms is of critical importance to many aspects of experimentation. Several features are implemented in order to limit horizontal aircraft velocity and acceleration (in the xy plane) so as to lessen vehicle aggression and corresponding noise disturbance. Additionally, maximal horizontal and vertical error distance components were established – the minimum permitted distance between the current and requested position before position fulfilment is deemed to have been successful. This parameter choice, however, was found to be very dependent on component wind forces – increased wind speeds dictate increased minimum error values.
- ***Experiment operation***: finally, operation of the full herd identity recovery experiment (as is described fully in Section 7.6) was exercised and validated with parameters relevant to the flight

testing location at the Fenswood Farm. To mimic live experimentation as closely as possible, 10 scale Holstein Friesian toy models were utilised such that efforts spent transitioning to real experiments were minimal. As such, the models were individually manually painted (see Figure 7.22) to form a realistic small herd population. Appropriate training data for respective detection and identification models was acquired via manual UAV flights at Fenswood Farm. Along with the $|R| = 10$ toy models, AprilTag fiducial markers were made visible [233, 321] allowing for automatic data labelling (for examples, see Figure 7.23) via transforming three dimensional toy bounding boxes into the camera plane via the process described in previous Chapter 5.4.1 on detector training data generation. The result of these validations meant that relevant models were simply retrained on labelled imagery of the real 17-strong population and, parameters were altered for operation at the Wyndurst Farm experiment location.

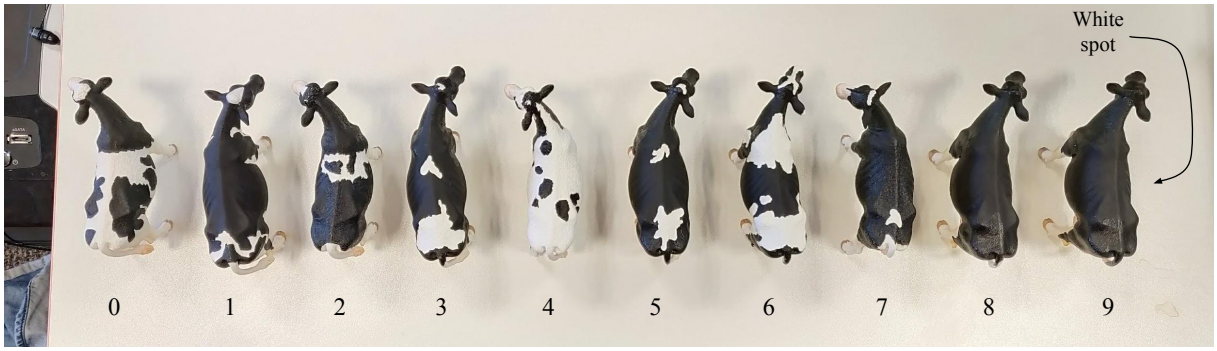


Figure 7.22: ***Individually-Unique Painted Cow Models.*** *Ten cow toy models were manually painted with individually-unique dorsal coat pattern markings to mimic randomly chosen individuals found in the datasets captured in earlier chapters. The small scale models with dimensions:* $5.1 \times 14.2 \times 8.1$ *cm were used to verify and validate experiment operation at the flight testing area at Fenswood Farm before conducting experiments on real cattle where welfare risks could potentially be introduced by system failure of untested setups.*

(a) $\sim$ *3m height above ground* *(b)* $\sim$ *5m height above ground*

Figure 7.23: ***Outdoor Acquired Training Data Generation.*** *(Top row): example frames from manual flights with targets automatically labelled via the use of the AprilTag fiducial marker system [233, 321]. Ground truth target annotations were then used to train models for visual detection and identification. The targets used are toy scale models of Holstein Friesian cattle utilised to verify experimental operation. Previous Figure 7.22 shows the painted models. (Bottom row): example auto-labelled individual RoIs at the stated heights above the ground illustrating the low resolutions of the toy cattle and the corresponding difficulty in detection and identification.*

7.6 Experiments, Results and Discussion

In this section, details, results and corresponding analyses and discussion points are given for experiments conducted in a real, live setting utilising the M100 UAV agent. This is organised, firstly, into the offline analysis of captured imagery, demonstrating the extent to which a single image evaluation paradigm is successful alongside corresponding limitations in Section 7.6.1. Subsequently, Section 7.6.2 provides preliminary results and findings for actual autonomous flights with online, live large-domain environment exploratory agency with individual-wise identity estimation occurring on-board the flight platform. Third, comprehensive illustrations and statistics are given for the analyses of conducted flight experiments in Section 7.6.3. Finally, a detailed discussion is given with respect to these experiments with suggestions for improvements in possible avenues of future work in Section 7.6.4.

As mentioned in the dataset acquisition stage (Section 7.3), the experimental period was carried out over a two week long session at the Wyndhurst Farm in Langford Village, UK. The first part of this period consisted of two days dealing entirely with data acquisition and acclimatising the cattle herd to the UAV. Following this, a brief period of data labelling and corresponding model training took place. Finally, and as this Section explores, experiments were carried out over three day long sessions with breaks between acceptable weather windows and UAV battery re-charging. To re-iterate, all flights were performed in accordance with CAA regulations for drones [16] alongside constraints imposed by the approved ethics application (UIN: UB/18/064).

Figure 7.24: ***Environment Coverage.*** *Screenshot of the DJI GO mobile application with past flight paths (white) overlaid for a small set of experiment flights. The flight paths correspond with example exploratory decisions depicted in Figure 7.34. Also indicated is the current controller position and orientation (red arrow) and the dynamically set home point (the GPS position the flight platform will attempt to land at in case of failure, low battery level, etc.). A smart device running the application is connected to the remote controller via USB providing telemetry, basic UAV control (e.g. land, return to home), a live camera feed and settings (e.g. exposure settings, capture resolution). Figure 7.1 illustrates relevant hardware in the employed setup and corresponding interactions.*

Figure 7.25: ***Experiment Photographs.*** *Photographs[26] from the conduction of real experiments at the Wyndhurst Farm in Langford Village, UK. (1^{st} row): the employed setup for experiment operation, (2^{nd} row): attempts (successful and not) to displace the herd towards the area of experiment operation, (3^{rd} row): the M100 flight platform in autonomous flight conducting live experiments and (4^{th} row): closeup images of the herd.*

[26] Many thanks to my father Tim Andrew for assisting with experimentation and capturing these very photographs.

7.6.1 Offline Model Performance Results

In this section, performance results for components implementing target detection and individual identification are given for *offline* operation and analysis. Towards this goal, all imagery captured during real, live flights is analysed here. The purpose being to assess these models on identical imagery in isolation from the constraint imposed by online experiments requiring targets to be central in the visual field before identification commences. Instead here, a concept of "attempt to detect and identify everything" is employed here to validate such a paradigm. Note however that throughout this section, single image analysis is performed for identification – as opposed to multi-iteration LRCN-based identification – due to data storage constraints on-board the UAV. Importantly, models, processes and respective parameters (neuron weights, model parameters etc.) were kept identical to the online scenario and are thus equivalent (except that the evaluation of multi-iteration identification is excluded here).

The dataset used for analysis here is the result of all *online* experiments across the three flight experiment day sessions. During every experiment, an acquired 720×720 image is saved to file at each exploratory agency iteration, yielding 1039 images across 18 autonomous flights, 99 of which actually contain targets/cows that were correspondingly labelled with ground truth bounding box annotations. Again, due to storage constraints on-board the UAV, only one of the $n = 5$ samples taken at each exploratory iteration is recorded to the on-board SD card, meaning that only single frame identification can be assessed offline. In the next section (7.6.2), improvements are described when operating iterative identification over the $n = 5$ samples online.

Tables 7.5 and 7.6 detail individual model performance across detection and identification components, respectively. To quantitatively conclude about component model performance, the 1039 instance strong image set was manually labelled including ground truth target bounding box annotations in accordance with VOC2012 labelling guidelines [92, 93] and corresponding identity labels. With respect to detection analysis, a detection was deemed a successful true positive provided sufficient IoU ($ov \geq 0.5$) for a predicted and ground truth bounding box given the binary classes: $\{cow, \neg cow\}$.

Descripiton	Value
True Positive	109
False Positive	7
False Negative	2
Accuracy (%)	**92.4**

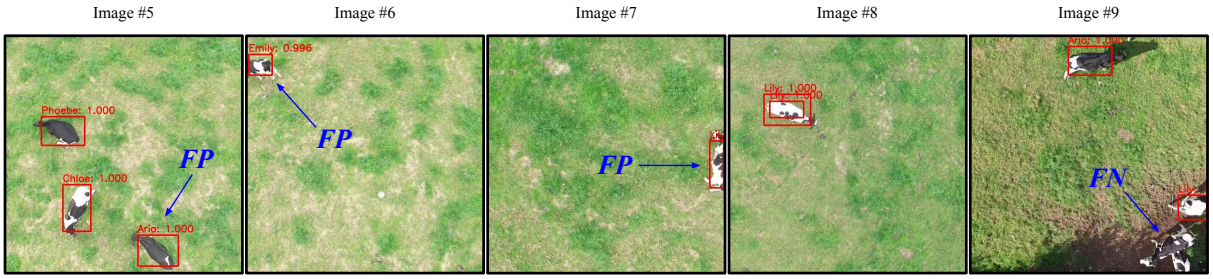
*Table 7.5: **Offline Detection Results.** Results for the offline detection and localisation of cattle targets from UAV-acquired imagery using a trained YOLO detector model. Positive detections are* $102/109 = 93.6\%$ *accurate, whilst overall accuracy including negatives is* 92.4%.

Figure 7.26 depicts examples of complete successes (detection and subsequent RoI identification) alongside failures in the identification and the detection models/processes individually. Strong illumination changes causing object shadows, changing contrast and image brightness as well as changing backgrounds can be seen across the image set, motivating the need for such variation across training examples in reality (e.g. multiple data acquisition/recording day sessions with varying weather conditions) and synthetically with image augmentation. In spite of these challenges, the process exhibits detection and identification robustness (as confirmed quantitatively in Tables 7.5 and 7.6) across success examples including partially occluded individuals (image #0), contrast (image #1) and viewpoint/distortion variation due to positional extremity (image #4) and more.

With respect to failures shown in Figure 7.26b, these can be seen to be caused by:

- ***Identification failure***: image #5 depicts an instance of pure identification model failure (false positive: Aria, ground truth: Evie), despite being provided a good RoI to be identified. As quanti-

(a) Detection and subsequent identification successes.



(b) Identification and detection failures.

Figure 7.26: ***Offline Result Examples.*** *Examples of detection and identification successes and failures from offline image analysis utilising identical models/weights for the online case executing on-board the UAV platform. Red boxes denote positive YOLO detections with sufficient confidence, text above denotes the predicted identity for that RoI alongside the model confidence in the respective label, whilst blue text indicates the type of failure.*

tatively demonstrated further in Table 7.6, Aria was the cause of multiple false positive identities as a result of spatially-situated dorsal visual similarities and also in structure to other individuals, Evie in particular (see Figure 7.6 for reference).

- ***Poor detection***: images #6 and #7 show marginal detection instances yielding partially occluded individuals and poor corresponding RoIs. These lead to to incorrect identity assessments as a result of the presence of partial dorsal feature visibility that aligns with other individuals more strongly.
- ***Multiple detections***: image #8 illustrates an example whereby the detection component indicated multiple nested RoIs for a single target, which was observed to occur somewhat infrequently; $6/118 = 5.08\%$ – further examples are shown in Figure 7.27. Whilst this is not actually a significant failure – especially since in all occurrences of this phenomenon, the correct identity was ultimately yielded – it was observed that the innermost detection typically had notably less confidence. As a result, this class of positive detections could be filtered out with a more carefully chosen threshold, perhaps at the cost of some valid true positive detections elsewhere.
- ***Detection failure***: image #9 depicts a case of pure detection failure (the bottom-right most individual) in which an individual was not detected. This occurs arguably as a result of the challenging illumination circumstance caused by a strong shadow cast by a high hedge/bush partially influencing the cow target itself in conjunction with a rare background (not grass). This particular lighting situation was a very rare observation that is equally difficult to augment training instances to account for.

ID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
Name	Olivia	Sophia	Amelia	Lily	Emily	Ava	Isabella	Mia	Ella	Poppy	Evie	Charlotte	Aria	Evelyn	Phoebe	Chloe	Daisy	**Totals**
True Positive	4	-	-	19	-	-	-	6	-	3	-	1	19	27	19	3	1	102
False Positive	-	-	-	1	1	-	-	-	-	-	-	-	5	-	-	-	-	7
Accuracy (%)	100			95	0			100		100		100	79.2	100	100	100	100	**93.6**

Table 7.6: ***Offline Identification Results.*** *Table of per-individual offline identification results and corresponding accuracies (overall accuracy:* 93.6%*) for the* 109 *detection instances on the original dataset of* $1039\ 720 \times 720$ *pixel images captured at each iteration throughout real online experiments. Accuracy for ID=12 (Aria) is demonstrably low because of visual similarities to ID=10 (Evie).*

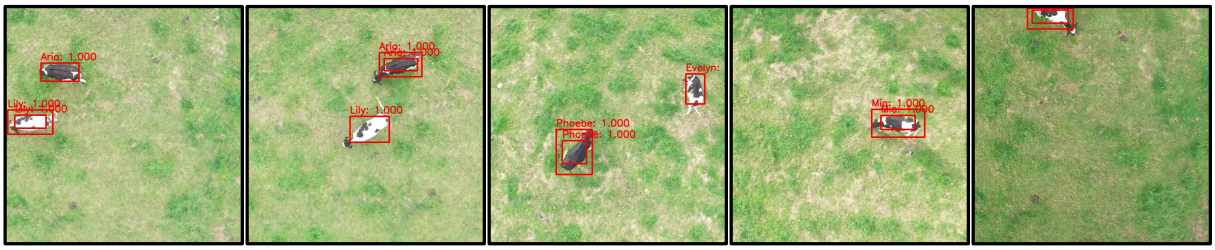


Figure 7.27: ***Double Target Detections.*** *Examples of the infrequent occurrence of double, nested target detections for offline image analysis. In all instances of this phenomenon, both generated RoIs given as input into the subsequent identification model yielded correct identities.*

	Loop Detected (%)	>1 Loops Detected (%)	>100 Moves (%)	Optimal Solution (%)	≤ 10 Difference (%)	$t > t_e$ Moves (%)	Target Recovery Rate ($^d/_t$)	Validation Accuracy (%)
pseudo-random *CU*	90.33	89.41	99.61	0.02	0.43	41.59	0.255 ± 0.057	72.45

Table 7.7: ***Synthetic/Offline Exploratory Agency Testing Results.*** *Performance statistics for exploratory agency operating offline across* $10,000$ *randomly-synthesised examples in pixel-image simulation for a* 20×20 *exploration environment with* 5×5 *agent-environment visibility attempting to discover* $|R| = 17$ *individuals. The importance is that this same trained model was utilised in real, online experiments operating on-board the UAV flight platform and the Jetson TX2 attempting to efficiently find new individuals to identify. Note that since the environment has considerably more possible grid locations than the original* 10×10 *size, maximal episode lengths are increased* $t_e = 400$ *to equal that number (*$20 \times 20 = 400$*). Refer back to Chapter 6 for full details on exploratory agency.*

7.6.2 Online Model Performance

This section presents results across conducted *online* experimental flights that were completely autonomous. That is, all computation was performed live in real time using the UAV's on-board computers (DJI Manifold & Nvidia Jetson TX2). Table 7.8 illustrates quantitative identification results across all conducted flight experiments. Example image sequences given as input to the LRCN identification pipeline – yielded from detections across multiple samples – are given in Figure 7.29, alongside corresponding model prediction and ground truth labels. Notably, across the small online sample size (18 actual instances), the model performs perfectly. This can be attributed to, firstly, the presence of multiple, varying individual-centric regions that allow the model to iteratively confirm identity as proved beneficial in earlier Chapter 4. Second, since this iterative identification process is only entered when a target is central within the agent's 5×5 visual field, the viewpoint is optimal with respect to dorsal animal visibility and pose (only the top of the cow is visible). Additionally, minimal image distortion from the camera lens is observed and the possibility of target clipping due to the frame boundary as present in the offline "detect and identify everything" case (see Figure 7.26b) is no longer an issue.

	LRCN Identification	Single Iteration Identification
# Samples	Accuracy (%)	Accuracy (%)
18	**100**	94.4

Table 7.8: ***Online Identification Results.*** *Performance statistics for the UAV agent performing online identification of targets via use of the LRCN-based passive iterative identification model on* 5 *frame-long RoI image sequences of individuals. Also included are results for normal single iteration identification evaluating the first frame in the* 5*-image sample sequence.*

To preliminarily conclude about whether an iterative identification process is beneficial here – with respect to a small herd size of 17 individuals with distinct visual uniqueness across the population (as can be inferred from Figure 7.6) – performance of a single iteration paradigm is compared here. Towards

this goal, the first frame of each of 18 image sequences was passed through the same identification estimation model. As can be seen in Table 7.8, a single iteration identification paradigm does not perform as accurately over the small sample size ($17/18 = 94.4\%$ instances predicted correctly). The single failure is exhibited as follows in Figure 7.28. As has been explored earlier in offline image assessment (Section 7.6.1), this single iteration identification paradigm suffers from a lack of sample variation and otherwise leading to losses in accuracy, particularly for individual Aria, as is the case for the single failure here.

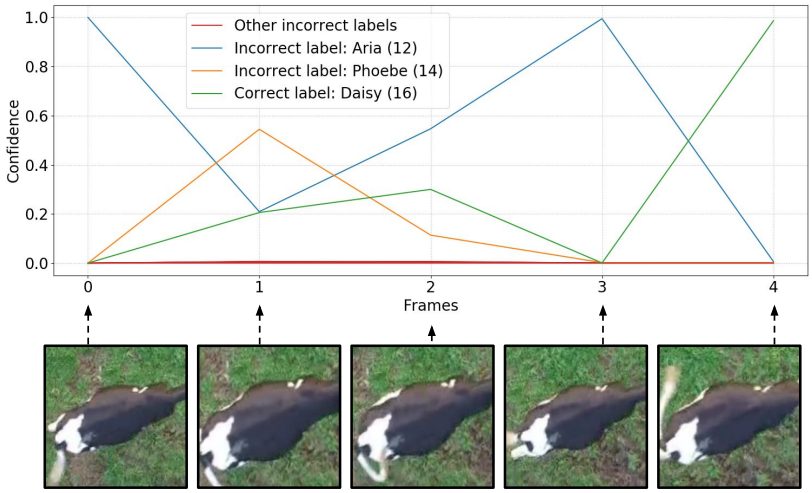


*Figure 7.28: **LRCN Model Confidence vs. Time (Frames).** Illustration of LRCN model confidence in the single correct and other incorrect identities versus exposure to subsequent frames. In this example, this figure depicts the single instance where the first frame alone was not sufficient to identify the individual correctly. In contrast, after significant model deliberation, $n = 5$ images correctly attribute the correct identity. Frames considered here qualitatively suggest that movement of the animal's tail led to discriminative feature occlusion yielding erroneous visual similarities comparable to Aria (ID=12). Slight model confidence in Phoebe results from similarities in an all black torso.*

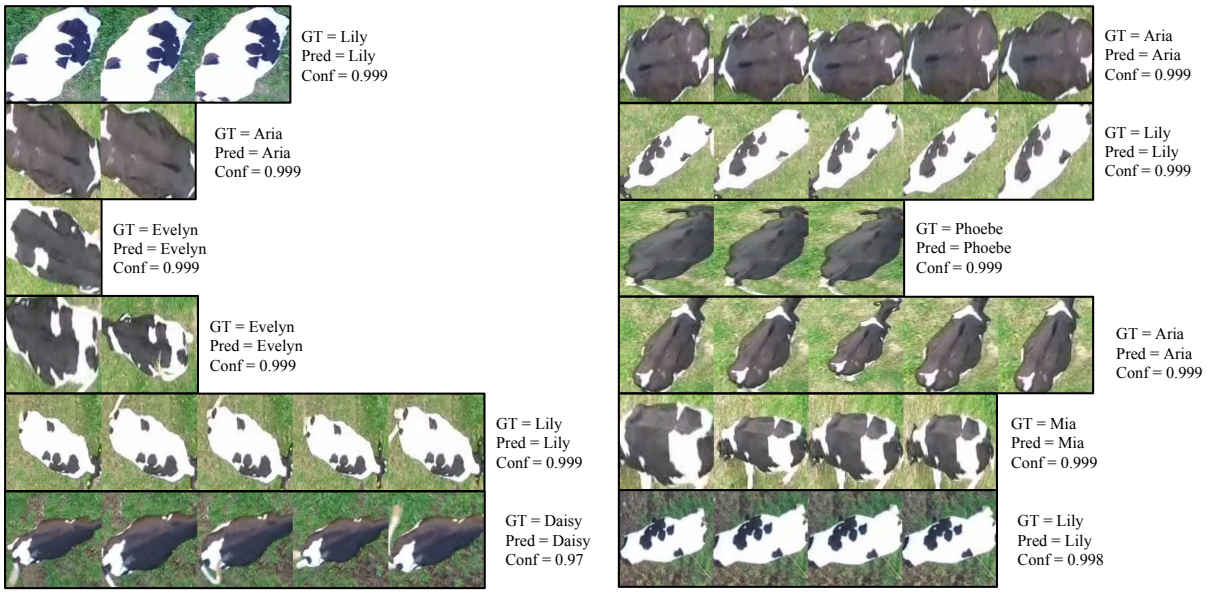


*Figure 7.29: **LRCN Identification Examples.** Twelve examples of 224×224 pixel RoI sequences presented as input into the passive LRCN-based identification pipeline alongside ground truth and predicted labels with associated respective model confidence values per instance. The maximum input length is $LSTM_{seq} = 5$ samples chosen to minimise flight time spent over an individual whilst allowing for some variation across frames from agent or target motion.*

7.6.3 Flight Analyses

This section gives a selection of figures providing individual and overall flight analyses. In particular, Table 7.9 gives detailed general statistics. Where relevant, metrics describing environment coverage quantify the percentage of the exploratory region that has been seen by the agent over the course of a flight. Note however, that considering this metric to conclusively quantify exploratory performance is only reasonable under the assumption that targets are completely static. In the case here, cattle move freely during experiments meaning previously visited/covered locations may now contain targets one wants to discover. Rather, the intention is for this metric to illustrate in how far the UAV battery can survive before landing is required. Also importantly, individual flight lengths were subject to variance – as can be seen in Figures 7.30, 7.31 and 7.32. This is by virtue of varying battery levels; not all batteries were consistently fully charged and, the exact elapsed time from system boot/initialisation to takeoff was also inconsistent. As well as this, experiments were sometimes manually cut short if there were no longer any targets present in the exploratory region (all cows had left the grid) and an empirical assessment determined that a target would not re-enter before the UAV battery depleted completely. The benefit of doing so lies in the operation of experiments where on location recharging batteries possibilities (in the field) is finite and therefore operationally costly.

Row	Description	Value
(a)	Number of experiment flights	18
(b)	Average time per exploration iteration	6.35s
(c)	Maximum iterations	98
(d)	Maximum flight time	10:16 mins
(e)	†Maximum environment coverage	85.75%
(f)	Median iterations	77
(g)	Median flight time	8:09 mins
(h)	†Median environment coverage	70.13%

Table 7.9: ***Flight Experiment Statistics.*** *Resulting flight and environmental statistics across the* 18 *conducted experimental flights. †: the percentage of exploratory map cells – for the large domain size consisting of* 20×20 *grid cells used here for real flights – that were visually observed over the course of the flight.*

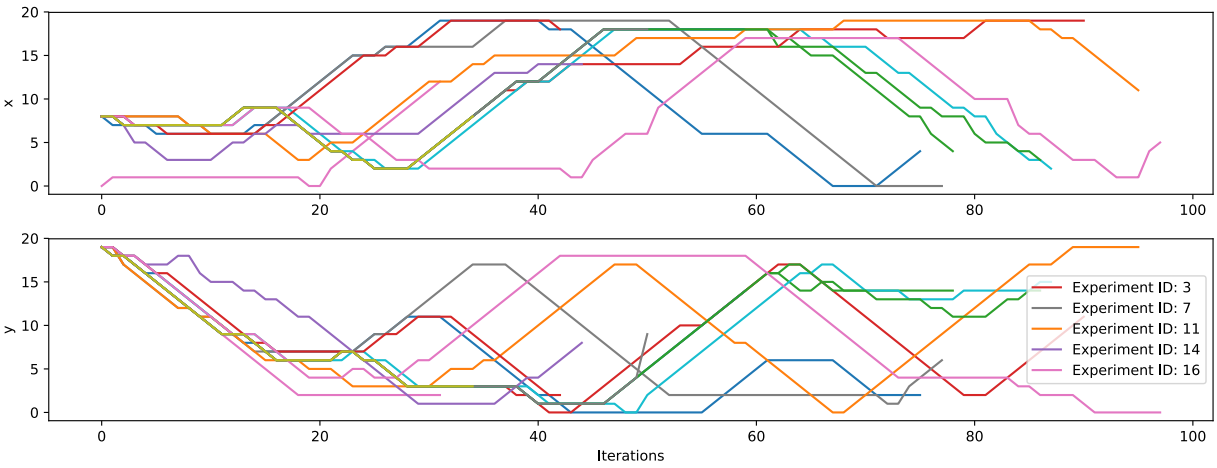
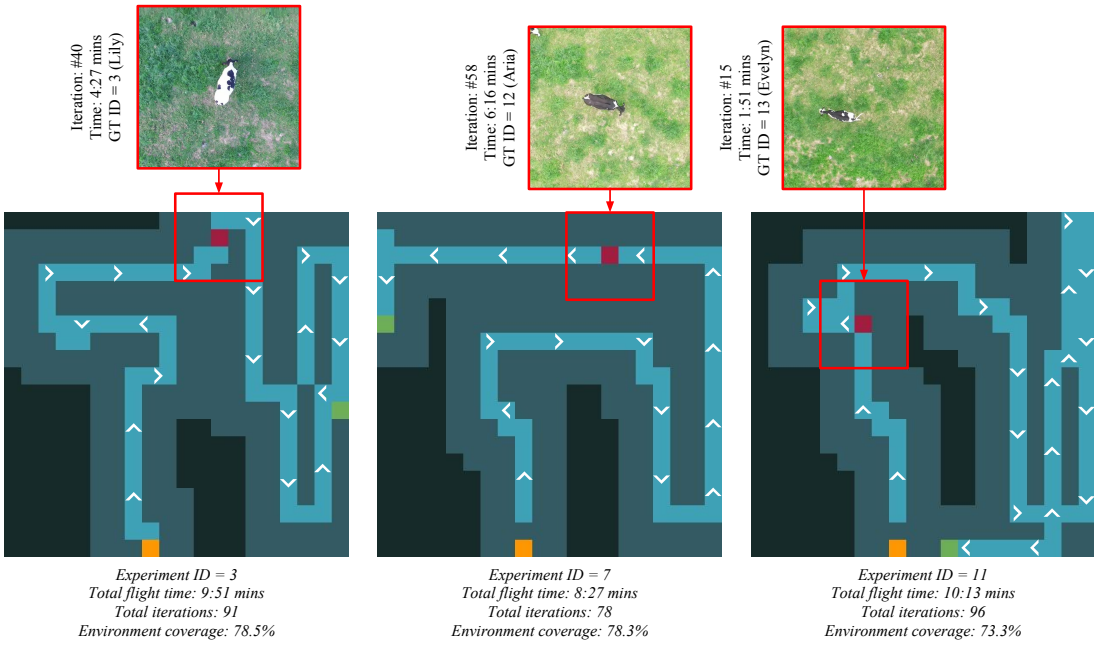
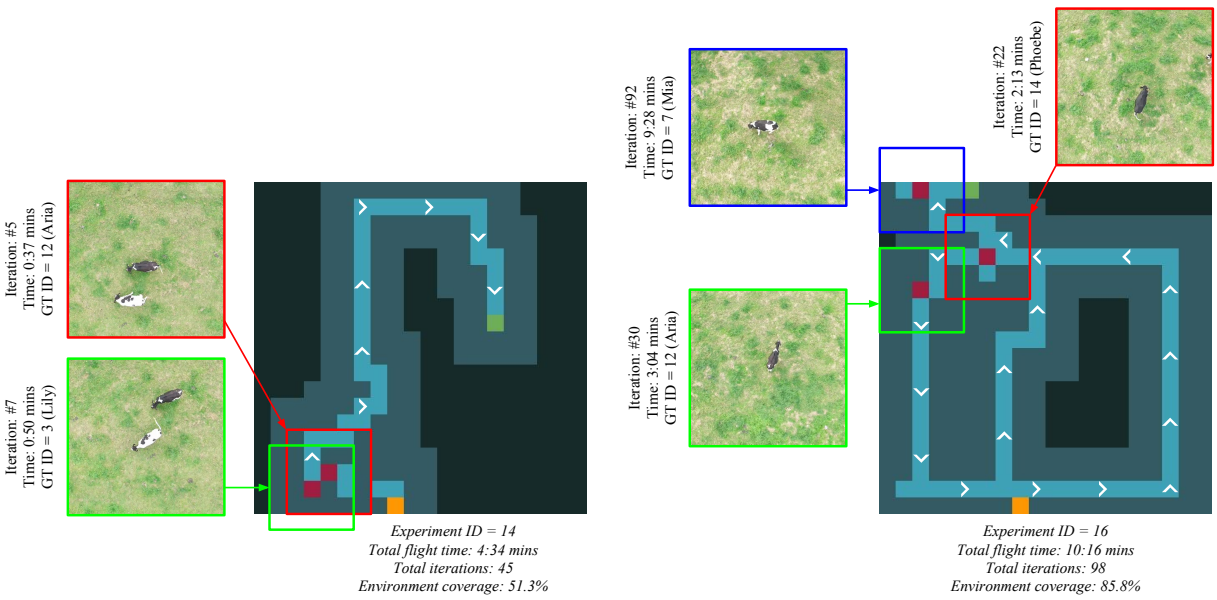


Figure 7.30: ***Agent Exploration Paths.*** *Illustration of agent* (x, y) *coordinates within the exploratory grid versus experiment iterations (time) over the* 18 *conducted experiment flights. Experiments shown in the legend are the highlighted examples shown in Figure 7.31. To complete 90 iterations, which equates to* $90/400 = 22.5\%$ *of possible exploratory cells, requires approximately* 10 *minutes of actual flight time.*

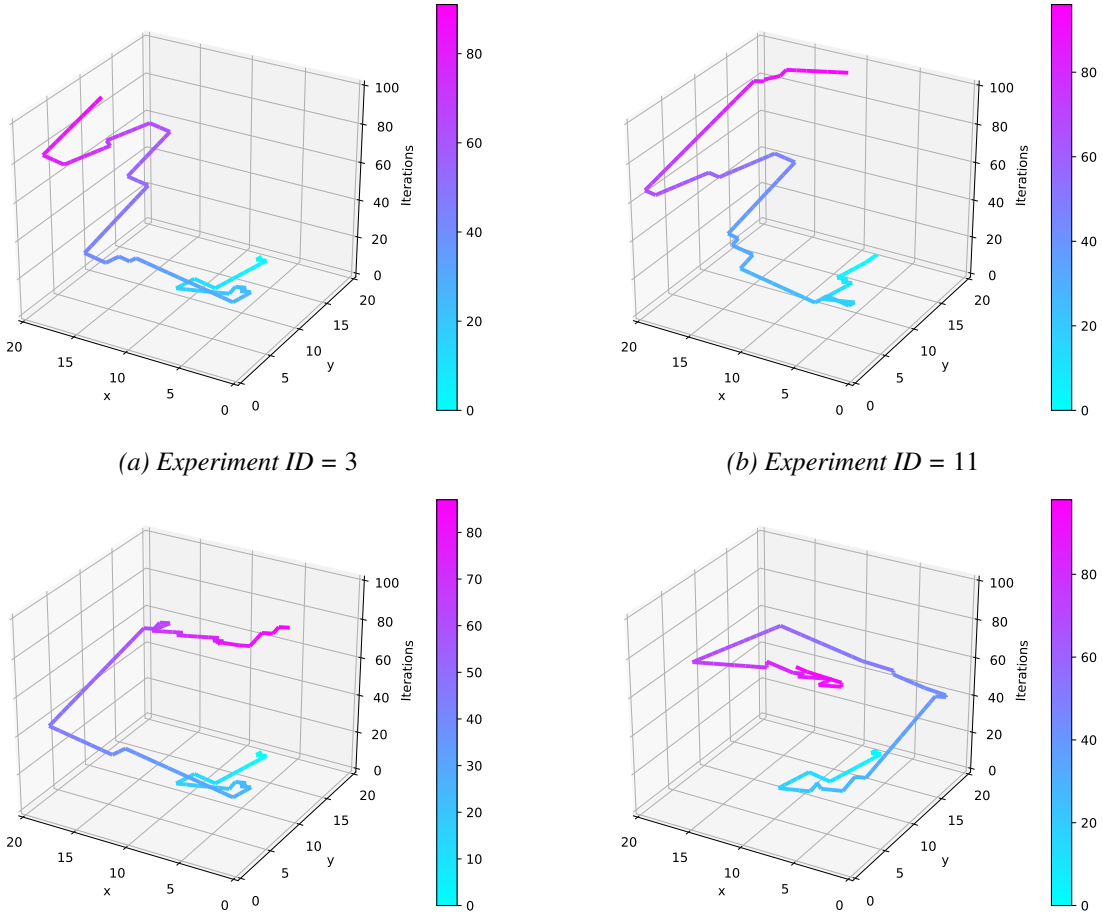


(a) Flights where a single individual was encountered and identified.



(b) Experiments with multiple individuals identified over the course of each flight.

Figure 7.31: ***Annotated Experiment Flight Paths.*** *Examples of annotated agent flight paths within the exploratory grid over the entire course of the experiment. In the context of the exploration grid, cell colours are defined as (black): unvisited locations, (light blue): visited locations, (dark blue): seen or covered locations (given the agent's* 5×5 *cell local environment visibility), (orange): agent starting position, (green): finishing agent position and (red): discovered target positions. At each target discovery point, local agent-environment visibility is indicated alongside the corresponding captured* 720×720 *image with statistics. Also illustrated – by white arrows – is the agent's direction of travel at iteration intervals throughout the experiment. Across every experiment, the average time required per iteration is:* $6.35s$*. The example flights shown here are highlighted in the legend of Figure 7.30.*



(a) Experiment ID = 3

(b) Experiment ID = 11

(c) Experiment ID = 12

(d) Experiment ID = 16

Figure 7.32: ***Exploratory Agency vs. Time.*** *Examples of agent-environment exploration paths x, y versus iterations (time) in the z-axis and a corresponding iteration colour gradient for illustrative purposes. As can be seen across this figure and others (see Figures 7.30 and 7.31), the initial exploratory paths are similar to begin with as a result of model determinism in the absence of target detection. Put differently, agent paths only diverge when knowledge of one or more target is gained, influencing future navigation decisions.*

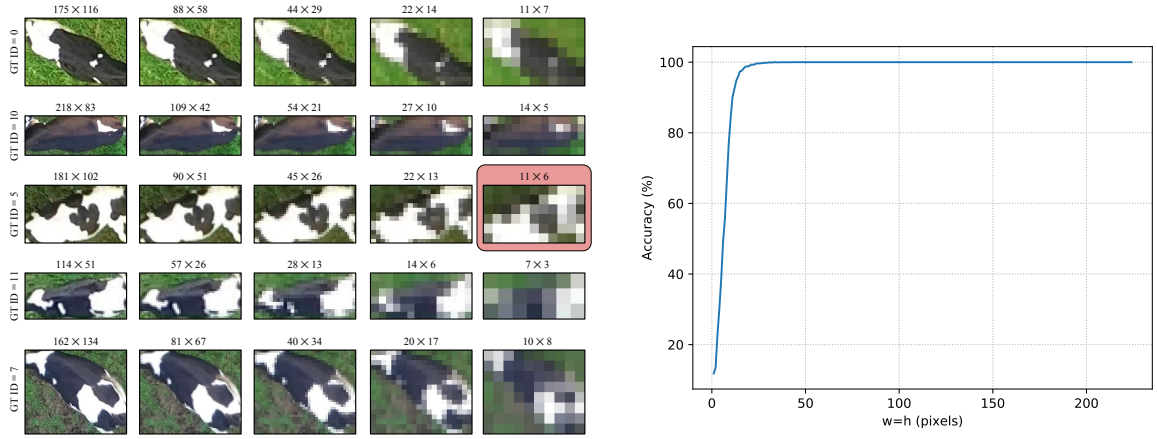
7.6.4 Discussion

This section provides discussion points for the preliminary experiments conducted in this chapter alongside suggestions for improvements within possible avenues of future work. To summarise, the limiting factors upon the real, live experiment were found to generally lie within elements outside of control, hardware and experimental setup choices rather than algorithmic choices, which bodes well for further work employing the techniques proposed in this work and their generalisation.

Environmental factors: An interesting discussion point is simply the fact that the experiment environment itself was simply imperfect. Principally, as a result of the field being situated alongside an immediately adjacent road, flight regulations do not permit $\leq$ 50m UAV proximity to structures and beings outside of human operator control [16]. Thus, experiments were unable to account for a majority proportion of the field and accordingly, the fitted exploratory grid includes just a small subset of the larger region. The result is that when the animals are faced with a choice of one area containing a loud, possibly threatening object in flight directly above them or another area without, natural instincts choose the perceived safer area of the field. In practise this meant that the animals would behave naturally (i.e. graze, sit) in the area of experimentation when the UAV was *not* present, and leave the area when it was. This being said, some individuals gradually became accustomed with agent presence, as will be discussed in the following paragraph.

Particular individuals: Despite considerable spent effort at the beginning of each day session in acclimatising the animals to the physical and loud sonic presence the UAV introduces in flight, it was found that this was ineffective on particular individuals. The result being that these individuals would more often that not walk away before the agent had sufficient time to collect identification samples, leave the experiment area of operation before being discovered by the agent or, not even enter the exploratory region altogether. Thus, there are some members of the population that were never discovered nor identified (as is reflected in Table 7.6) meaning conclusions regarding identification performance across the entire population are impossible to draw. Conversely, individuals that were more comfortable with the presence of the UAV were therefore more likely to remain under the agent when in flight meaning that multiple samples – consisting of exploratory agency discovery, target detection and iterative identification – are obtained. Possible solutions to this problem: spending more time in animal acclimatisation or more easily, flying at a higher altitude where the sound level will decrease – reasons for why this simple solution was not employed are given in the following paragraph.

Camera/image resolution: Ideally, to improve aforementioned animal comfort as well as environment visibility from an exploratory standpoint, the UAV would be flown at a higher selected height above the ground. The result of doing so would mean that individual target resolution decreases proportionally with increased height and thus, both detection and identification components would operate on lower resolution features. Whilst perhaps not a problem for species-wide detection (as proven in the reliable detection of small toy cows in Section 7.5.5), identification accuracy will suffer as a result of a potentially insufficient sampling frequency for dorsally exhibited spatial patterns, structures and alignments. Figure 7.33 demonstrates that this drop-off in model accuracy occurs at a very low resolution of 25×25 pixels in the case here operating on a small $|R| = 17$ herd size. In the generalisation of this model to larger herd sizes where, the likelihood of similar-looking individuals is higher, the resolution threshold at which accuracy falls will increase.



(a) *(b)*

Figure 7.33: ***Individual RoI Resolution.*** *Visualisation of (a): varying individual-wise RoI resolutions illustrating the difficulty in individual identification when presented lower resolutions. Each of the examples above are from the training dataset – and thus, the network had seen previously – were evaluated by the identity estimation network, the single incorrect prediction is highlighted in red. The graph in (b): demonstrates the effect on model prediction accuracy whilst increasing the resolution of the input given as input into the predictor. The graph highlights that lower resolution inputs are successfully predicted across the small 17-strong herd population where there are little intra-population visual similarities. Again, note that identification accuracy here was considered on examples from the training dataset and in reality, this result would likely worsen.*

The inherent trade-off with this consideration lies in increasing environment visibility/coverage and hence quicker target position discoveries versus decreasing identification accuracy and thus overall herd identity recovery. Parameter balance was established empirically such that targets are *at minimum* resolved to $\sim 150 \times 150$ pixels at a flight height above the ground of 10m. Then conveniently, this selected

height corresponds to almost exact $10m^2$ local visibility (centred about the agent) which was chosen to equate to 5×5 exploration map coverage given the camera's horizontal field of view for image resolution 720×720. This is valid under the assumption of constant camera resolution, which is the case here – images can only be obtained at a maximum resolution of 960×720, and this image is then trimmed square to 720×720. The clear solution to both problems would be performing flight at a higher altitude with a zoom lens or higher resolution camera. However with height increase the potential for image blurring from camera gimbal imperfection, higher wind speeds, etc. is attributable.

Target distribution class: The exploratory agency component performed live in reality within this chapter was trained on pseudo-random target positional distribution (utilising the Mersenne Twister P-RNG algorithm [208]) with $|R| = 17$ targets within a 20×20 exploratory environment. However, this is arguably not a realistic approximation of the herd's actual distribution; it was observed that the animals would sometimes clump together closely, particularly when perhaps uncomfortable with the presence of the UAV. Other times, the animals would naturally graze freely as one would expect, with seemingly no discernible distribution; justifying training against pseudo-random examples. The problem is then that, since the agent sometimes directly affects the herd's behaviour, it is incredibly difficult to model or approximate an appropriate distribution class exploratory agency can be trained against. Solutions lie in a more complex formation of a dynamic target distribution model and improving animal-robot acclimatisation.

Entering and leaving: Following on from the discussion in the previous paragraph, another incorrect assumption within the target distribution class approximation (that was trained against) is simply that the exploratory region has no boundary in reality. As a result, animals are free to enter and leave the exploratory environment autonomously, whereas the model was trained against the assumption that all $|R| = 17$ targets are situated somewhere within map bounds. The issue is that this affects exploratory strategies when there is inter-target correlation for some positional distributions, therefore excluding fully random target distributions. This also affects the ability to assess the performance of exploratory agency within the scope of this chapter, since there is no ground-truth knowledge of which individuals are within the environment over experiment time or exploration iterations. The obvious solution to this problem lies within the experiment environment itself being imperfect; other more suitable fields would allow UAV flight throughout, or a physical barrier could be erected around the exploratory region such that targets cannot leave. An alternative approach could model dynamic distribution classes whereby targets can freely enter and leave the environment.

Exploration iteration time: In relation to the exploratory component of experiments conducted here, the time required by the UAV controller to fulfil a requested position involves significant overheard (on average, 6.35s per exploration iteration). Since exploratory actions (and resulting agent movements) arrive in the form of discrete, one grid cell displacements per iteration, the result is that the flight platform spends a significant proportion of the total flight time attempting to realise goal positions. This inefficiency is perhaps misspent time; of which there is little due to payload mass and power constraints. This problem is then magnified by the choice of large environment sizes that are necessary to resolve a single target to a single grid cell. In the case here – operating on a 40×40m environment with $2m^2$ grid cells yielding a 20×20 exploratory grid – 400 possible grid positions to visit is infeasible given the reality of battery constraints imposing $\sim$10 minute flights and therefore, $\sim$100 possible iterations. Instead, the more important metric is the extent to which the agent covers the exploratory region visually or how much has the agent seen? Accordingly, across all 18 conducted experiments, the median coverage of the exploratory grid was approximately 70.13%[27], as is depicted in Figure 7.34. Whilst this value is specifically relevant under an assumption of static targets, given target dynamism and autonomy (as is the case for real cattle), the agent must sometimes revisit previously seen locations, and the small number ($\sim$100) of possible exploratory iterations calls for concern on the average exploratory iteration time and its resulting implications. It is at this point interesting to note the difference in the design of algorithms for operation

[27]This value is approximate because aircraft batteries were not always charged completely equally and platform landing was typically imposed manually, thus varying across flights.

in a perfect world (in simulation) in comparison to constraints imposed by operational reality. Related suggestions for future work ameliorating this problem specifically are given in Section 8.3.

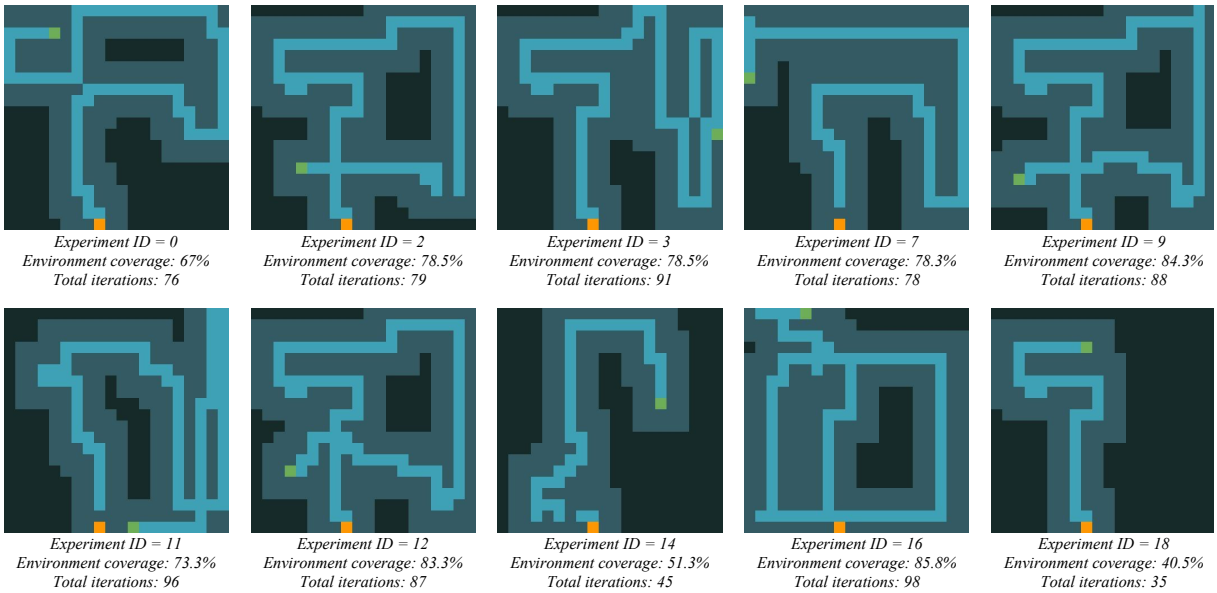


Figure 7.34: ***Environment Explorations.*** *Examples of environment exploration paths and coverage over* 10 *real flight experiments – of varying lengths – on the fitted* 20×20 *exploratory grid with agent local visibility of* 5×5 *cells for a* 720×720 *pixel image operating at a height of 10m. Note that the determinism of the exploratory agency architecture is visible within instances where no target was detected; the bottom-right most exploration (experiment ID* $= 18$*) depicts this baseline movement pattern. Note also the presence across some instances of seemingly impossible path gaps. This phenomenon is caused by slight position fulfilment errors (particularly in high winds) due to UAV localisation as a result of inherent GPS inaccuracy. As will be explored, this problem could be alleviated by a more accurate positioning system (e.g. RTK-GPS system [143]).*

Exploration grid resolution: With respect to exploratory agency – in transitioning from the simulated case to operation in reality – particular assumptions or approximations become inherently flawed, as was observed in earlier discussion points. In the case here, there is an assumption that a target should be perfectly resolved to a particular exploratory grid cell, when in fact it could situated perfectly in-between two positions. The result could entail perceived failure to centre the individual within the image for identification estimation to actually take place due to camera distortion. The problem lies in the decision to assume a discrete two-dimensional model of what is intrinsically a continuous world. Solutions could involve increasing the exploratory grid resolution (the sampling frequency) such that targets occupy multiple grid cells or assuming continuous agent and target positions (related propositions are discussed further in Section 8.3).

UAV height/altitude maintenance: Across flights conducted at both locations, it was found that the UAV exhibited significant positional error in the vertical dimension, especially under high wind forces. This error was observed to fluctuate massively (approximately $\pm$1m) during position fulfilment and also whilst hovering at a commanded position. The result is that visibility of the environment changes across iterations and accordingly, samples captured at each iteration. Consequences of such variation impact subsequent image analysis components; boundary targets may be detected that otherwise would not have been (and vice versa), individuals may be insufficiently resolved for correct identity analysis, etc. Solutions to this problem lie within the flight platform's chosen method of localisation – the M100 utilises GPS, explaining the exhibited inaccuracy (particularly in the vertical dimension [25, 137]). Instead, a more accurate localisation methodology could be employed; RTK-GPS, a satellite navigation technique used to enhance GPS precision providing centimetre-level position accuracy [143].

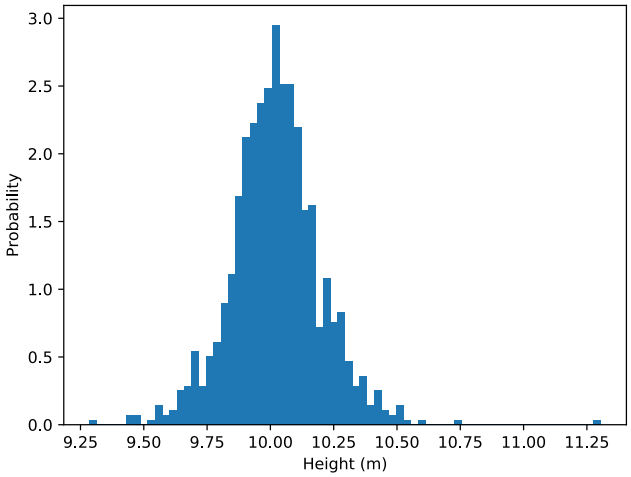


Figure 7.35: ***UAV Height Distribution.*** *Distribution of the UAV height above the ground over* $\sim 1{,}000$ *samples with min*=9.28*, max*=11.31*,* $\mu = 10.02$*,* $\sigma = 0.18$ *following a Gaussian-like distribution curve. The commanded height value was* 10*m. Note that each instance actually consists of an average UAV local position value over* $n = 5$ *samples to allow the controller to settle, gather multiple images, etc. Thus, this distribution is not necessarily truly representative meaning UAV height maintenance and fluctuation is actually worse in reality.*

7.7 Chapter Conclusion

This chapter demonstrates a proof-of-concept that individual cattle identities can be reliably recovered live by a robot agent in a real-world agricultural setting. As part of this demonstration, several components described in previous chapters are algorithmically and operationally verified in reality here; primarily the ability for relevant models to meaningfully integrate complementary imagery in live scenarios, and second, the exhibition of online exploratory agency towards recovering herd-like distributions. In robustly performing these tasks on-board a UAV-based robot with limited computational resources alongside payload, weight restrictions and more, the proposed technologies serve as validation into future agricultural automation possibilities.

Chapter 8

Conclusion

8.1 Summary

This thesis demonstrates that coat patterns exhibited by the prevalent Holstein Friesian or "dairy cow" cattle species provide a sufficient basis for individual identification across small population sizes (e.g. < 100) that are commonplace in real farms [6, 5]. Built upon the assumption of dorsal coat patterns being an individually-unique biometric entity, this work proposes several visual biometric processes that exploit such features.

Chapter 3 – Single Frame Identification: to begin, a preliminary demonstration that dorsal features are appropriate for individual identification is given by assessing single images. Achieved via use of classical hand-crafted features, robust identification accuracy of 96.6% is achieved over a small 25-strong population. Use of the ASIFT feature descriptor, however, involves significant computational expense in extracting, characterising and matching image keypoints. This burden is ameliorated by a contemporary approach founded in the use of artificial neural networks, specifically convolutional architectures, achieving 86% mAP on a broader set of 89 individuals. A key separation being that the supervised learning-based approach operates on closed population sets, whereas components of the proposed local feature matching pipeline generalise to unseen individuals.

Chapter 4 – Passive Multi-Frame Identification: to that point, the proposed biometric processes dealt with the evaluation of a single image from an aerial perspective. However, as motivated by the subtlety of discriminative features for fine-grained object categories, small perturbations in image acquisition, object viewpoint, object autonomy, light conditions, etc. can prevent observations of crucial features. In capturing and assessing multiple images, where these small perturbations occur, the likelihood of an observation containing important features increases. This intuition is demonstrated for a passive agent with no influence on observation parameters, where accuracy is found to generally increase versus exposure to multiple variant frames that complement identity estimates. This is found to be the case across small time windows (i.e. 1-2s), where limited variation across frames actually takes place. Quantitatively speaking, this multi-frame biometric process ultimately yielded 98.13% identification accuracy on a testing set of 40 frame-long image sequences, whereas assessing the first frame in each sequence only achieved 80.47%.

Chapter 5 – Simulated Active Multi-Frame Identification: building upon this identification agency, a unified model performing active identification is shown to produce robust and efficient results on herds simulated in a realistic three-dimensional environment. The method consists of a distinct training phase where object-wise viewpoints are exhaustively searched for containing visual descriptions of individuality. This form of behaviour is then replicated online by a simulated flying robot agent actively seeking viewpoints for a perceived object identity that disambiguates its class to a satisfactory degree as quickly as possible. Over the full experiment consisting of ten difficult synthetic identification cases, the pro-

posed pipeline correctly identifies the individual in question 60.34% of the time when an active multi-frame identification approach is required and enacted. This chapter concludes the proposal of biometric processes performing identification of a single individual the agent is spatially local to.

Chapter 6 – Simulated Inter-Individual Navigation: armed with the ability to identify individuals, consideration of finding the remaining members of the herd – the collection of individuals – is necessary. Accordingly, exploration strategies that exhaustively search every environment location, amongst other implemented baselines, are shown to be outperformed by a single deep framework that co-optimises local sensing and global environment knowledge under a unified architecture. The ability to efficiently discover the positions of unknown targets is demonstrated to expand to a wide variety of target distributions, dynamics and behaviours.

Chapter 7 – Proof-of-Concept: Real-World Herd Individual Identification: finally, components are combined into a single framework performing passive iterative identification of targets found by the proposed exploratory agency algorithm. This framework operates on-board a real UAV with associated restrictions on computational power. Experiments are conducted on a collection of 17 live cattle with demonstrable identification robustness (100% accuracy across 18 samples) for the small herd in a real-world outdoor agricultural environment. Performing flights in reality identified limitations of the proposed exploratory algorithm, largely focussing around exploration time and the employed approximative model of the environment. Accordingly, suggestions for improvement are given in the following section on future work.

8.2 Discussion

In consideration of automatically identifying individuals in-barn, the setup employed in Chapter 3 (a fixed camera acquiring an aerial view of a nearly single file walkway) demonstrated promising results. Algorithmically speaking, both proposed methods displayed problems with the detection of multiple cattle as one, by virtue of their close proximity and disruptive camouflage. Importantly, this occurred in an area of the barn that constricts space to only allow few individuals in the camera frame at any one time. Were the camera to be affixed elsewhere (at potentially more open areas), the likelihood of this problem occurring would certainly increase. Spending effort therefore, on improving inter-individual separation and detection components is crucial in achieving robust camera location-agnostic identification performance.

With respect to autonomous UAV-based identification operating over fields containing cattle, refer to the discussions in Section 7.6.4 and below in Section 8.3 for operational considerations going forward. Echoing the comments made in Section 1.1.2, from a biological, veterinarian and behavioural standpoint, interesting next steps in this experiment lie in individual tracking over time. Whereby analysis of such data could inform upon grazing patterns [116, 117], social hierarchies [313, 166, 243], herd welfare [294] and more. A key improvement should however, revolve around further minimising the disturbance to the animals – thereby minimising influence on their behaviour – by modifying the UAV platform itself for reducing noise production [290, 205], operating at higher altitudes, etc.

In its entirety however, this work provides a strong case that the automated identification of Holstein Friesian cattle is viable, robust and efficient within the proposed visual biometric processes. In doing so, a proof-of-concept is given for removing the necessity for permanent and damaging animal tagging methods, promising to improve animal welfare in husbandry.

8.3 Future Work

In this section, potential avenues of future work are suggested as follows. Immediate further work will concentrate on the preparation of the content presented in Chapter 7 for publication.

Exploration grid discretisation: the exploratory grid for agent navigation is approximated in this work to be a discrete set of coordinates, when in reality, this involves careful, user-selected choice of parameters when fitting a grid to an environment. Future work could involve investigating modelling the environment continuously, such that the problem becomes a question of two-dimensional regression. In this form, discrete 2m jumps are no longer made iteratively, and unexplored areas of the environment can be quickly explored; an important factor when presented with limited UAV battery life. In addition, the problem of targets being positioned directly between two grid cells is intrinsically avoided altogether.

Live Active Identification Experiments: whilst Chapter 5 validates the proposed active identification algorithm in realistic three-dimensional simulations, conducting experiments in reality with a live UAV agent is a logical progression. These experiments could be performed across several stages *(1)*: on static fake targets indoors, where agent localisation can be perfectly resolved thanks to motion capture camera systems, *(2)*: the same experiment but outdoors where position variation is introduced due to GPS, *(3)*: on real (and now moving) cattle examined individually and finally *(4)*: integrating environment exploration and active individual identification (when necessary) on a full-sized herd.

The core challenges involving firstly; acquiring and hand-labelling significant imagery per-viewpoint per-individual such that *(a)*: the space can be effectively searched offline for good trajectories and *(b)* constituent models of the active identification architecture have sufficient amounts and variety of labelled training data. Second, inaccuracies exhibited by models estimating pose, distance and scale via vision are compounded by agent localisation error from GPS, hence the suggestion of stage *(1)* where this effect can be separated. And finally, the introduction of target dynamism involves significant challenge when latency from processing and/or realising a goal position now renders that decision invalid (e.g. by the time the agent moves to position x to view the left side of an individual, the cow has moved away). Looking beyond the specific task of identifying individual cattle, applications of the proposed time-sensitive active viewpoint ordering algorithm using a mobile robot could involve solving (with adaptation) the kidnapped robot problem [97] (seeking to observe the presence or absence of visual landmarks to disambiguate robot location), pick & place objectives mandating object category determination with a robot arm and more.

Environment visibility: as a result of agent-environment exploratory actions only being performed in the horizontal xy-plane, the agent's local visibility of the environment from its on-board camera is always fixed. There are situations when this is the only option (e.g. indoor environments), however, it would be interesting to investigate the benefit of involving the additional z dimension in exploratory strategies. One would imagine the agent initially taking a high, global view of the environment, and subsequently focusing attention by reducing its height, much like attentional mechanisms in two-dimensional imagery for fine-grained categories [98].

Category scalability: throughout this work, the finite set of object categories has had relatively low cardinality (< 100), in accordance with typical herd sizes and populations in real farms. Investigations into model scalability would be interesting from a vision perspective in investigating how well the proposed methodologies generalise to large amounts of fine-grained categories. In addition, it would be biologically compelling to infer the relationship between population size and exploitable uniqueness amongst exhibited coat patterns.

Category adaptability: throughout this thesis, possible categories have been fixed across experiments, corresponding to the population of individuals being constant. When in actuality, farms often add or takeaway sets of individuals from a field or barn environment. The result is that newly-added individuals will be falsely identified, and subtracted individuals introduce surplus categories when the problem has

lowered in complexity. There is correspondingly, a need for model robustness and adaptability on variation in category cardinality. Future work could investigate automatically indicating newly-introduced individuals and applying zero-shot learning [331] or some registration process on the new, unseen fine-grained categories.

Multi-robot or multi-sensor approach: as has been discussed previously, a pitfall of flight operation in a horizontal plane only means that the scope of local agent-centric sensing is always fixed. Whilst this could be influenced with the addition of a flight height component, an alternate approach could involve a dual robot scenario or additional sensing (a zoom camera/gimbal). The idea being that one robot or sensor maintains a global view of the environment and constituent targets, performing multi-object tracking. This entity then indicates to another robot or sensor when a particular individual needs (re)identification, such that fine-grained local identification can take place.

Operational reality: were the full UAV-based herd monitoring system to be deployed in reality, maximum flight lengths of 10 minutes are quite limiting and may not satisfy experiment intentions (e.g. social monitoring). Instead, a UAV fleet is conceivable with offset cycles of $\{fly, recharge\}$ such that continual environment coverage is achieved. Work in this area could focus on (a): swarm-based path planning to avoid collisions, (b): automatic hand-off of data from a robot going to recharge to another taking over and (c): autonomous flying robot battery recharging.

Outside of this specific extension, effort should be spent on maximising the work achieved with a single battery/flight by considering bottlenecks. One key area to do with environment exploration has already been discussed (see "Exploration grid discretisation" above). The next involves the fact that at each iteration, multiple data samples are taken of vehicle position, attitude and camera image. These samples are uniformly acquired over a time window such that values are permitted to settle (position and attitude) and sufficient variation in imagery is advantageous when estimating identity via the LRCN. This creates an obvious inefficiency that could be alleviated by firstly: employing a more accurate robot localisation scheme (e.g. RTK-GPS [143]) in combination with more finely-tuned UAV controller parameters such that multiple pose samples are unnecessary. Second, as established evidentially in Section 7.6, and indeed in many other parts of this thesis, well-acquired single images often provide sufficient discriminative spatial information. Enforcing LRCN-based identification at every request is perhaps unnecessary and could be retained for cases where the model is unsatisfactorily confident in some identity in order to improve efficiency per iteration.

References

[1] A. Abd-Elrahman, L. Pearlstine, and F. Percival. Development of pattern recognition algorithm for automatic bird detection from unmanned aerial vehicle imagery. *Surveying and Land Information Science*, 65(1):37, 2005.

[2] B. Adler, J. Xiao, and J. Zhang. Finding next best views for autonomous uav mapping through gpu-accelerated particle simulation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1056–1061. IEEE, 2013.

[3] S. Agatonovic-Kustrin and R. Beresford. Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, 22(5):717–727, 2000.

[4] G. Aggarwal, A. R. Chowdhury, and R. Chellappa. A system identification approach for video-based face recognition. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 4, pages 175–178. IEEE, 2004.

[5] Agriculture and H. D. Board. Dairy statistics: An insider's guide 2015. 2015.

[6] Agriculture and H. D. Board. Farm data - average size of dairy herds. 2017.

[7] E. Ahmed, M. Jones, and T. K. Marks. An improved deep learning architecture for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3908–3916, 2015.

[8] S. Ahmed, T. Gaber, A. Tharwat, A. E. Hassanien, and V. Snáel. Muzzle-based cattle identification using speed up robust feature approach. In *Intelligent Networking and Collaborative Systems (INCOS), 2015 International Conference on*, pages 99–104. IEEE, 2015.

[9] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze. Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. *IEEE Robotics & Automation Magazine*, 19(3):80–91, 2012.

[10] A. Allen, B. Golden, M. Taylor, D. Patterson, D. Henriksen, and R. Skuce. Evaluation of retinal imaging technology for the biometric identification of bovine animals in northern ireland. *Livestock Science*, 116(1):42–52, 2008.

[11] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, M. Hasan, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

[12] W. Andrew, C. Greatwood, and T. Burghardt. Visual localisation and individual identification of holstein friesian cattle via deep learning. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2850–2859, Oct 2017.

[13] M. Andriluka, S. Roth, and B. Schiele. Monocular 3d pose estimation and tracking by detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 623–630. IEEE, 2010.

[14] A. C. Arslan, M. Akar, and F. Alagoz. 3d cow identification in cattle farms. In *Signal Processing and Communications Applications Conference (SIU), 2014 22nd*, pages 1347–1350. IEEE, 2014.

[15] E. Atkins. Autonomy as an enabler of economically-viable, beyond-line-of-sight, low-altitude uas applications with acceptable risk. In *AUVSI unmanned Systems*, 2014.

[16] C. A. Authority. The drone code. `http://dronesafe.uk/wp-content/uploads/2018/06/Dronecode_2018-07-30.pdf`. [Online; accessed 26-July-2018].

[17] A. I. Awad. From classical methods to animal biometrics: A review on cattle identification and tracking. *Computers and Electronics in Agriculture*, 123:423–435, 2016.

[18] A. I. Awad, A. E. Hassanien, and H. M. Zawbaa. A cattle identification approach using live captured muzzle print images. In *Advances in Security of Information and Communication Networks*, pages 143–152. Springer, 2013.

[19] A. I. Awad, H. M. Zawbaa, H. A. Mahmoud, E. H. H. A. Nabi, R. H. Fayed, and A. E. Hassanien. A robust cattle identification scheme using muzzle print images. In *Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on*, pages 529–534. IEEE, 2013.

[20] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, 33(8):1619–1632, 2011.

[21] J. E. Banta, L. Wong, C. Dumont, and M. A. Abidi. A next-best-view system for autonomous 3-d object reconstruction. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 30(5):589–598, 2000.

[22] Y. M. Bar-On, R. Phillips, and R. Milo. The biomass distribution on earth. *Proceedings of the National Academy of Sciences*, page 201711842, 2018.

[23] A. Baranov, R. Graml, F. Pirchner, and D. Schmid. Breed differences and intra-breed genetic variability of dermatoglyphic pattern of cattle. *Journal of animal breeding and genetics*, 110(1-6):385–392, 1993.

[24] B. Barry, U. Gonzales-Barron, K. McDonnell, F. Butler, and S. Ward. Using muzzle pattern recognition as a biometric approach for cattle identification. *Transactions of the ASABE*, 50(3):1073–1080, 2007.

[25] M. Barth and J. A. Farrell. The global positioning system & inertial navigation. *McGraw-Hill*, 8:21–56, 1999.

[26] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417, 2006.

[27] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[28] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[29] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *IEEE transactions on pattern analysis and machine intelligence*, 33(9):1806–1819, 2011.

[30] J. A. Berni, P. J. Zarco-Tejada, L. Suárez, and E. Fereres. Thermal and narrowband multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle. *IEEE Transactions on Geoscience and Remote Sensing*, 47(3):722–738, 2009.

[31] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart. Receding horizon" next-best-view" planner for 3d exploration. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1462–1468. IEEE, 2016.

[32] P. S. Blaer and P. K. Allen. Data acquisition and view planning for 3-d modeling tasks. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 417–422. IEEE, 2007.

[33] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2017.

[34] F. Bonin-Font, A. Ortiz, and G. Oliver. Visual navigation for mobile robots: A survey. *Journal of intelligent and robotic systems*, 53(3):263, 2008.

[35] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots in cluttered environments. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 572–577. IEEE, 1990.

[36] J. Borenstein and Y. Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE transactions on robotics and automation*, 7(3):278–288, 1991.

[37] H. Borotschnig, L. Paletta, M. Prantl, and A. Pinz. Appearance-based active object recognition. *Image and Vision Computing*, 18(9):715–727, 2000.

[38] H. Borotschnig, L. Paletta, M. Prantl, A. Pinz, et al. Active object recognition in parametric eigenspace. In *BMVC*, pages 1–10. Citeseer, 1998.

[39] M. Bowling, D. Pendell, D. Morris, Y. Yoon, K. Katoh, K. Belk, and G. Smith. Identification and traceability of cattle in selected countries outside of north america. *The Professional Animal Scientist*, 24(4):287–294, 2008.

[40] S. Branson, G. Van Horn, S. Belongie, and P. Perona. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952*, 2014.

[41] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1515–1522. IEEE, 2009.

[42] B. Browatzki, V. Tikhanoff, G. Metta, H. H. Bülthoff, and C. Wallraven. Active object recognition on a humanoid robot. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2021–2028. IEEE, 2012.

[43] W. Buick. Animal passports and identification. *Defra Veterinary Journal*, 15:20–26, 2004.

[44] T. Burghardt. *Visual animal biometrics: automatic detection and individual identification by coat pattern*. PhD thesis, University of Bristol, 2008.

[45] C. Cai and J. Li. Cattle face recognition using local binary pattern descriptor. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*, pages 1–4. IEEE, 2013.

[46] V. Caporale, A. Giovannini, C. Di Francesco, and P. Calistri. Importance of the traceability of animals and animal products in epidemiology. *Revue Scientifique et Technique-Office International des Epizooties*, 20(2):372–378, 2001.

[47] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[48] D. Chen, Z. Yuan, B. Chen, and N. Zheng. Similarity learning with spatial constraints for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1268–1277, 2016.

[49] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1335–1344, 2016.

[50] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[51] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *Proceedings of the IEEE international conference on computer vision*, pages 3029–3037, 2015.

[52] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, pages 1610–02357, 2017.

[53] S. Chowdhury, M. Marufuzzaman, H. Tunc, L. Bian, and W. Bullington. A modified ant colony optimization algorithm to solve a dynamic traveling salesman problem: A case study with drones for wildlife surveillance. *Journal of Computational Design and Engineering*, 2018.

[54] P. Christiansen, K. A. Steen, R. N. Jørgensen, and H. Karstoft. Automated detection and recognition of wildlife using thermal cameras. *Sensors*, 14(8):13778–13793, 2014.

[55] S.-J. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar. A survey on aerial swarm robotics. *IEEE Transactions on Robotics*, 34(4):837–855, 2018.

[56] J. R. Clynch. Earth coordinates. *Electronic Documentation, February*, 2006.

[57] A. F. Cobo and F. C. Benıtez. Approach for autonomous landing on moving platforms based on computer vision. 2016.

[58] V. Codreanu, F. Dong, B. Liu, J. B. Roerdink, D. Williams, P. Yang, and B. Yasar. Gpu-asift: A fast fully affine-invariant feature extraction algorithm. In *High Performance Computing and Simulation (HPCS), 2013 International Conference on*, pages 474–481. IEEE, 2013.

[59] C. Connolly. The determination of next best views. In *Robotics and automation. Proceedings. 1985 IEEE international conference on*, volume 2, pages 432–435. IEEE, 1985.

[60] I. C. Cuthill, M. Stevens, J. Sheppard, T. Maddocks, C. A. Párraga, and T. S. Troscianko. Disruptive coloration and background pattern matching. *Nature*, 434(7029):72, 2005.

[61] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[62] N. Daucher, M. Dhome, J.-T. Lapresté, and G. Rives. Modelled object pose estimation and tracking by monocular vision. In *BMVC*, volume 93, pages 249–258. Citeseer, 1993.

[63] B. L. Decker. World geodetic system 1984. Technical report, Defense Mapping Agency Aerospace Center St Louis Afs Mo, 1986.

[64] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *ICRA*, volume 2, pages 1322–1328, 1999.

[65] C. Deng, S. Wang, Z. Huang, Z. Tan, and J. Liu. Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications. *J. Commun*, 9(9):687–692, 2014.

[66] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[67] F. Department for Environment and R. Affairs. *The Cattle Book 2008: Descriptive statistics of cattle numbers in Great Britain on 1 June 2008*. DEFRA, 2008.

[68] G. N. DeSouza and A. C. Kak. Vision for mobile robot navigation: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 24(2):237–267, 2002.

[69] P. D'Hour, A. Hauwuy, J. Coulon, and J. Garel. Walking and dairy cattle performance. In *Annales de zootechnie*, volume 43, pages 369–378, 1994.

[70] J. J. DiCarlo, D. Zoccolan, and N. C. Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

[71] S. J. Dickinson, H. I. Christensen, J. K. Tsotsos, and G. Olofsson. Active object recognition integrating attention and viewpoint control. *Computer vision and image understanding*, 67(3):239–260, 1997.

[72] S. Ding, L. Lin, G. Wang, and H. Chao. Deep feature learning with relative distance comparison for person re-identification. *Pattern Recognition*, 48(10):2993–3003, 2015.

[73] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[74] M. Dorigo and L. M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation*, 1(1):53–66, 1997.

[75] M. Dougoud, C. Mazza, B. Schwaller, and L. Pecze. Extending the mathematical palette for developmental pattern formation: Piebaldism. *arXiv preprint arXiv:1710.03563*, 2017.

[76] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T.-K. Kim. Recovering 6d object pose and predicting next-best-view in the crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3583–3592, 2016.

[77] DroneDeploy. DroneDeploy: powerful cloud-based drone software. `https://dronedeploy.com/`, 2016. [Online; accessed 2-November-2016].

[78] S. Edelman. *Representation and Recognition in Vision*. MIT Press, Cambridge, MA, USA, 1999.

[79] D. Edwards and A. Johnston. Welfare implications of sheep ear tags. *The Veterinary Record*, 144(22):603–606, 1999.

[80] D. Edwards, A. Johnston, and D. Pfeiffer. A comparison of commonly used ear tags on the ear damage of sheep. *Animal Welfare*, 10(2):141–151, 2001.

[81] H. M. El-Bakry, I. El-Hennawy, and H. M. El Hadad. Bovines muzzle identification using box-counting. *International Journal of Computer Science and Information Security*, 12(5):29, 2014.

[82] H. M. El Hadad, H. A. Mahmoud, and F. A. Mousa. Bovines muzzle classification based on machine learning techniques. *Procedia Computer Science*, 65:864–871, 2015.

[83] I. El-Henawy, H. El-bakry, and H. El-Hadad. Muzzle classification using neural networks. *International Arab Journal of Information Technology (IAJIT)*, 14(4), 2017.

[84] I. El-Henawy, H. M. El Bakry, and H. M. El Hadad. Cattle identification using segmentation-based fractal texture analysis and artificial neural networks. *International Journal of Electronics and Information Engineering*, 4(2):82–93, 2016.

[85] I. El-Henawy, H. M. El Bakry, and H. M. El Hadad. A new muzzle classification model using decision tree classifier. *International Journal of Electronics and Information Engineering*, 6(1):12–24, 2017.

[86] I. El-Henawy, H. M. El Hadad, and N. Mastorakis. Muzzle feature extraction based on gray level co-occurrence matrix. *International Journal of Veterinary Medicine*, 1:16–24, 2016.

[87] H. Enting, D. Kooij, A. Dijkhuizen, R. Huirne, and E. Noordhuizen-Stassen. Economic losses due to clinical lameness in dairy cattle. *Livestock production science*, 49(3):259–267, 1997.

[88] J. B. Escario, J. F. Jimenez, and J. M. Giron-Sierra. Ant colony extended: experiments on the travelling salesman problem. *Expert Systems with Applications*, 42(1):390–410, 2015.

[89] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. Robust multiperson tracking from a mobile platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1831–1846, 2009.

[90] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2012 (voc2012) results (2012). In *URL http://www. pascal-network. org/challenges/VOC/voc2011/workshop/index. html*, 2011.

[91] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2007 (voc 2007) results (2007), 2008.

[92] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[93] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[94] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani. Person re-identification by symmetry-driven accumulation of local features. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2360–2367. IEEE, 2010.

[95] D. B. Fogel. An evolutionary approach to the traveling salesman problem. *Biological Cybernetics*, 60(2):139–144, 1988.

[96] G. Fosgate, A. Adesiyun, and D. Hird. Ear-tag retention and identification methods for extensively managed water buffalo (bubalus bubalis) in trinidad. *Preventive veterinary medicine*, 73(4):287–296, 2006.

[97] D. Fox, S. Thrun, W. Burgard, and F. Dellaert. Particle filters for mobile robot localization. In *Sequential Monte Carlo methods in practice*, pages 401–428. Springer, 2001.

[98] J. Fu, H. Zheng, and T. Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[99] T. Gaber, A. Tharwat, A. E. Hassanien, and V. Snasel. Biometric cattle identification approach based on webers local descriptor and adaboost classifier. *Computers and Electronics in Agriculture*, 122:55–66, 2016.

[100] J. Gasteiger and J. Zupan. Neural networks in chemistry. *Angewandte Chemie International Edition in English*, 32(4):503–527, 1993.

[101] E. Gavves, B. Fernando, C. G. Snoek, A. W. Smeulders, and T. Tuytelaars. Fine-grained categorization by alignments. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1713–1720, 2013.

[102] A. P. Gay, T. P. Stewart, R. Angel, M. Easey, A. J. Eves, N. J. Thomas, D. A. Pearce, and A. I. Kemp. Developing unmanned aerial vehicles for local and flexible environmental and agricultural monitoring. In *Proceedings of RSPSoc 2009 Annual Conference. RSPSoc*, pages 471–476, 2009.

[103] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao. Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search. *Applied Soft Computing*, 11(4):3680–3689, 2011.

[104] F. A. Gers and J. Schmidhuber. Recurrent nets that time and count. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, volume 3, pages 189–194. IEEE, 2000.

[105] S. Getzin, R. S. Nuske, and K. Wiegand. Using unmanned aerial vehicles (uav) to quantify spatial gap patterns in forests. *Remote Sensing*, 6(8):6988–7004, 2014.

[106] S. Getzin, K. Wiegand, and I. Schöning. Assessing biodiversity in forests using very high-resolution images and unmanned aerial vehicles. *Methods in Ecology and Evolution*, 3(2):397–404, 2012.

[107] J.-H. Giraldo-Zuluaga, A. Salazar, A. Gomez, and A. Diaz-Pulido. Automatic recognition of mammal genera on camera-trap images using multi-layer robust principal component analysis and mixture neural networks. *arXiv preprint arXiv:1705.02727*, 2017.

[108] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[109] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[110] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2016.

[111] G. B. Goh, N. O. Hodas, and A. Vishnu. Deep learning for computational chemistry. *Journal of computational chemistry*, 38(16):1291–1307, 2017.

[112] Y. Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[113] S. Gong, M. Cristani, C. C. Loy, and T. M. Hospedales. The re-identification challenge. In *Person re-identification*, pages 1–20. Springer, 2014.

[114] L. Green, V. Hedges, Y. Schukken, R. Blowey, and A. Packington. The impact of clinical lameness on the milk yield of dairy cows. *Journal of dairy science*, 85(9):2250–2256, 2002.

[115] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 2016.

[116] P. Gregorini. Diurnal grazing pattern: its physiological basis and strategic management. *Animal Production Science*, 52(7):416–430, 2012.

[117] P. Gregorini, S. Tamminga, and S. Gunter. Behavior and daily grazing patterns of cattle. *The Professional Animal Scientist*, 22(3):201–209, 2006.

[118] K. D. Gremban and K. Ikeuchi. Planning multiple observations for object recognition. *International journal of computer vision*, 12(2-3):137–172, 1994.

[119] G. Grisetti, C. Stachniss, W. Burgard, et al. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34, 2007.

[120] C. Groba, A. Sartal, and X. H. Vázquez. Solving the dynamic traveling salesman problem using a genetic algorithm with trajectory prediction: An application to fish aggregating devices. *Computers & Operations Research*, 56:22–32, 2015.

[121] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation. *arXiv preprint arXiv:1702.03920*, 2017.

[122] T. Gurriet and L. Ciarletta. Towards a generic and modular geofencing strategy for civilian uavs. In *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*, pages 540–549. IEEE, 2016.

[123] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on signal processing*, 50(2):425–437, 2002.

[124] M. F. Hansen, M. L. Smith, L. N. Smith, K. A. Jabbar, and D. Forbes. Automated monitoring of dairy cow body condition, mobility and weight using a single 3d video capture device. *Computers in industry*, 98:14–22, 2018.

[125] M. F. Hansen, M. L. Smith, L. N. Smith, M. G. Salter, E. M. Baxter, M. Farish, and B. Grieve. Towards on-farm pig face recognition using convolutional neural networks. *Computers in Industry*, 98:145–152, 2018.

[126] M. Hausknecht and P. Stone. Deep recurrent q-learning for partially observable mdps. *CoRR, abs/1507.06527*, 2015.

[127] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer, 2014.

[128] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[129] M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962.

[130] C. S. Helvig, G. Robins, and A. Zelikovsky. The moving-target traveling salesman problem. *Journal of Algorithms*, 49(1):153–174, 2003.

[131] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[132] S. Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1), 1991.

[133] S. Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Master's thesis, Institut fur Informatik, Technische Universitat, Munchen*, 1991.

[134] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[135] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[136] A. Hodgson, N. Kelly, and D. Peel. Unmanned aerial vehicles (uavs) for surveying marine fauna: a dugong case study. *PloS one*, 8(11):e79556, 2013.

[137] B. Hofmann, H. Lichtenegger, and J. Collins. Gps theory and practice. *Springer Wien NewYork*, 2001.

[138] Holstein Foundation. *Working With Dairy Cattle*. Holstein Foundation, 2009.

[139] B. Horne, M. Jamshidi, and N. Vadiee. Neural networks in robotics: A survey. *Journal of Intelligent and Robotic Systems*, 3(1):51–66, 1990.

[140] R. Houston. A computerised database system for bovine traceability. *Revue Scientifique et Technique-Office International des Epizooties*, 20(2):652, 2001.

[141] Z.-C. Huang, X.-L. Hu, and S.-D. Chen. Dynamic traveling salesman problem based on evolutionary computation. In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, volume 2, pages 1283–1288. IEEE, 2001.

[142] X. Hui, J. Bian, X. Zhao, and M. Tan. Vision-based autonomous navigation approach for unmanned aerial vehicle transmission-line inspection. *International Journal of Advanced Robotic Systems*, 15(1):1729881417752821, 2018.

[143] H. P. Intelligence. Real-time kinetic - an introduction to gnss. `https://www.novatel.com/an-introduction-to-gnss/chapter-5-resolving-errors/real-time-kinematic-rtk/`. [Online; accessed 15-Oct-2018].

[144] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[145] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza. An information gain formulation for active volumetric 3d reconstruction. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 3477–3484. IEEE, 2016.

[146] M. Israel. A uav-based roe deer fawn detection system. *International Archives of Photogrammetry and Remote Sensing*, 38:1–5, 2011.

[147] Itseez. Open source computer vision library. `https://github.com/itseez/opencv`, 2015.

[148] K. A. Jabbar, M. F. Hansen, M. Smith, and L. Smith. Overhead spine arch analysis of dairy cows from three-dimensional video. In *Eighth International Conference on Graphic and Image Processing (ICGIP 2016)*, volume 10225, page 102250E. International Society for Optics and Photonics, 2017.

[149] K. A. Jabbar, M. F. Hansen, M. L. Smith, and L. N. Smith. Early and non-intrusive lameness detection in dairy cows using 3-dimensional video. *Biosystems Engineering*, 153:63–69, 2017.

[150] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[151] A. Johnston, D. Edwards, E. Hofmann, P. Wrench, F. Sharples, R. Hiller, W. Welte, and K. Diederichs. 1418001. welfare implications of identification of cattle by ear tags. *The Veterinary Record*, 138(25):612–614, 1996.

[152] F. Jolai and A. Ghanbari. Integrating data transformation techniques with hopfield neural networks for solving travelling salesman problem. *Expert Systems with Applications*, 37(7):5331–5335, 2010.

[153] D. Jones. Power line inspection-a uav concept. In *Autonomous Systems, 2005. The IEE Forum on (Ref. No. 2005/11271)*, pages 8–pp. IET, 2005.

[154] G. P. Jones. *The feasibility of using small unmanned aerial vehicles for wildlife research*. PhD thesis, University of Florida, 2003.

[155] S. D. Jones, C. Andresen, and J. L. Crowley. Appearance based process for visual navigation. In *Intelligent Robots and Systems, 1997. IROS'97., Proceedings of the 1997 IEEE/RSJ International Conference on*, volume 2, pages 551–557. IEEE, 1997.

[156] G. P. JONES IV, L. G. Pearlstine, and H. F. Percival. An assessment of small unmanned aerial vehicles for wildlife research. *Wildlife Society Bulletin*, 34(3):750–758, 2006.

[157] R. Jozefowicz, W. Zaremba, and I. Sutskever. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2342–2350, 2015.

[158] Z. Kalal, K. Mikolajczyk, J. Matas, et al. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409, 2012.

[159] N. Kalchbrenner, I. Danihelka, and A. Graves. Grid long short-term memory. *arXiv preprint arXiv:1507.01526*, 2015.

[160] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–II. IEEE, 2004.

[161] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, volume 2, page 1, 2011.

[162] A. Kimura, K. Itaya, and T. Watanabe. Structural pattern recognition of biological textures with growing deformations: A case of cattle's muzzle patterns. *Electronics and Communications in Japan (Part II: Electronics)*, 87(5):54–66, 2004.

[163] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[164] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2149–2154. IEEE.

[165] L. P. Koh and S. A. Wich. Dawn of drone ecology: low-cost autonomous aerial vehicles for conservation. *Tropical Conservation Science*, 5(2):121–132, 2012.

[166] S. Kondo and J. Hurnik. Stabilization of social hierarchy in dairy cows. *Applied Animal Behaviour Science*, 27(4):287–297, 1990.

[167] A. Kosaka and A. C. Kak. Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties. *CVGIP: Image understanding*, 56(3):271–329, 1992.

[168] W. R. Koski, T. Allen, D. Ireland, G. Buck, P. R. Smith, A. M. Macrander, M. A. Halick, C. Rushing, D. J. Sliwa, and T. L. McDonald. Evaluation of an unmanned airborne system for monitoring marine mammals. *Aquatic Mammals*, 35(3):347, 2009.

[169] E. Kramer, D. Cavero, E. Stamer, and J. Krieter. Mastitis and lameness detection in dairy cows by application of fuzzy logic. *Livestock Science*, 125(1):92–96, 2009.

[170] J. Krause, H. Jin, J. Yang, and L. Fei-Fei. Fine-grained recognition without part annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5546–5555, 2015.

[171] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[172] H. S. Kühl and T. Burghardt. Animal biometrics: quantifying and detecting phenotypic appearance. *Trends in ecology & evolution*, 28(7):432–441, 2013.

[173] S. Kumar, A. Pandey, K. S. R. Satwik, S. Kumar, S. K. Singh, A. K. Singh, and A. Mohan. Deep learning framework for recognition of cattle using muzzle point image pattern. *Measurement*, 116:1–17, 2018.

[174] S. Kumar and S. K. Singh. Feature selection and recognition of muzzle point image pattern of cattle by using hybrid chaos bfo and pso algorithms. In *Advances in Chaos Theory and Intelligent Control*, pages 719–751. Springer, 2016.

[175] S. Kumar and S. K. Singh. Visual animal biometrics: survey. *IET Biometrics*, 6(3):139–156, 2016.

[176] S. Kumar and S. K. Singh. Automatic identification of cattle using muzzle point pattern: a hybrid feature extraction and classification paradigm. *Multimedia Tools and Applications*, 76(24):26551–26580, 2017.

[177] S. Kumar, S. K. Singh, and A. K. Singh. Muzzle point pattern based techniques for individual cattle identification. *IET Image Processing*, 2017.

[178] S. Kumar, S. K. Singh, R. Singh, and A. K. Singh. Animal biometrics: Concepts and recent application. In *Animal Biometrics*, pages 1–20. Springer, 2017.

[179] S. Kumar, S. K. Singh, R. S. Singh, A. K. Singh, and S. Tiwari. Real-time recognition of cattle using animal biometrics. *Journal of Real-Time Image Processing*, 13(3):505–526, 2017.

[180] S. Kumar, S. Tiwari, and S. K. Singh. Face recognition of cattle: can it be done? *Proceedings of the National Academy of Sciences, India Section A: Physical Sciences*, 86(2):137–148, 2016.

[181] I. Kuzovkin, R. Vicente, M. Petton, J.-P. Lachaux, M. Baciu, P. Kahane, S. Rheims, J. R. Vidal, and J. Aru. Activations of deep convolutional neural networks are aligned with gamma band activity of human visual cortex. *Communications biology*, 1, 2018.

[182] S. Kyristsis, A. Antonopoulos, T. Chanialakis, E. Stefanakis, C. Linardos, A. Tripolitsiotis, and P. Partsinevelos. Towards autonomous modular uav missions: The detection, geo-location and landing paradigm. *Sensors*, 16(11):1844, 2016.

[183] M. A. Leaders, V. S. Calculator, A. C. A. Network, and A. Store. Welfare implications of hot-iron branding and its alternatives. *Backgrounder*, 2011.

[184] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[185] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[186] V. Lepetit, P. Fua, et al. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends® in Computer Graphics and Vision*, 1(1):1–89, 2005.

[187] Y. Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

[188] Y. Li, S. Gong, and H. Liddell. Constructing facial identity surfaces for recognition. *International Journal of Computer Vision*, 53(1):71–92, 2003.

[189] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *European conference on computer vision*, pages 15–29. Springer, 2012.

[190] Z. Li, Y. Liu, R. Hayward, J. Zhang, and J. Cai. Knowledge-based power line detection for uav surveillance and inspection systems. In *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference*, pages 1–6. IEEE, 2008.

[191] S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2197–2206, 2015.

[192] C. F. Liew, D. DeLatte, N. Takeishi, and T. Yairi. Recent developments in aerial robotics: An survey and prototypes overview. *arXiv preprint arXiv:1711.10085*, 2017.

[193] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[194] L. Lin and M. A. Goodrich. Uav intelligent path planning for wilderness search and rescue. In *Intelligent robots and systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 709–714. IEEE, 2009.

[195] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[196] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[197] T.-Y. Lin, A. RoyChowdhury, and S. Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1449–1457, 2015.

[198] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[199] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[200] H. A. Mahmoud and H. M. R. E. Hadad. Automatic cattle muzzle print classification system using multiclass support vector machine. *International Journal of Image Mining*, 1(1):126–140, 2015.

[201] M. Malmir, K. Sikka, D. Forster, I. Fasel, J. R. Movellan, and G. W. Cottrell. Deep active object recognition by joint label and action prediction. *Computer Vision and Image Understanding*, 156:128–137, 2017.

[202] M. Malmir, K. Sikka, D. Forster, J. R. Movellan, and G. Cottrell. Deep q-learning for active recognition of germs: Baseline performance on a standardized dataset for active learning. In *BMVC*, pages 161–1, 2015.

[203] P. Martinet, N. Daucher, J. Gallice, and M. Dhome. Robot control using monocular pose estimation. *Worshop on New Trends in Image-based Robot Servoing, IROS*, 97:1–12, 1997.

[204] C. A. Martinez-Ortiz, R. M. Everson, and T. Mottram. Video tracking of dairy cows for assessing mobility scores. 2013.

[205] K. Massey and R. Gaeta. Noise measurements of tactical uavs. In *16th AIAA/CEAS aeroacoustics conference*, page 3911, 2010.

[206] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[207] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda. Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks*, 16(5-6):555–559, 2003.

[208] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[209] Y. Matsumoto, M. Inaba, and H. Inoue. Visual navigation using view-sequenced route representation. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 83–88. IEEE, 1996.

[210] D. Meagher. Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147, 1982.

[211] A. A. Melnikov, A. Makmal, and H. J. Briegel. Projective simulation applied to the grid-world and the mountain-car problem. *arXiv preprint arXiv:1405.5459*, 2014.

[212] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. Von Stryk. Comprehensive simulation of quadrotor uavs using ros and gazebo. In *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, pages 400–411. Springer, 2012.

[213] W. T. Miller, P. J. Werbos, and R. S. Sutton. *Neural networks for control*. MIT press, 1995.

[214] P. Milner, K. Page, and J. Hillerton. The effects of early antibiotic treatment following diagnosis of mastitis detected by a change in the electrical conductivity of milk. *Journal of dairy science*, 80(5):859–863, 1997.

[215] H. Minagawa, T. Fujimura, M. Ichiyanagi, and K. Tanaka. Identification of beef cattle by analyzing images of their muzzle patterns lifted on paper. *Publications of the Japanese Society of Agricultural Informatics*, 8:596–600, 2002.

[216] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

[217] S. Mishra, D. O. O. D. C. BREEDING, and R. NO. *Studies on the characteristics of muzzle dermatoglyphics in dairy cattle and buffalo*. PhD thesis, NDRI, 1994.

[218] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[219] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[220] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[221] L. Moisan and B. Stival. A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. *International Journal of Computer Vision*, 57(3):201–218, 2004.

[222] H. P. Moravec. The stanford cart and the cmu rover. *Proceedings of the IEEE*, 71(7):872–884, 1983.

[223] J.-M. Morel and G. Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[224] S. S. Mousavi, M. Schukat, and E. Howley. Deep reinforcement learning: an overview. In *Proceedings of SAI Intelligent Systems Conference*, pages 426–440. Springer, 2016.

[225] M. Müller. Computer go. *Artificial Intelligence*, 134(1-2):145–179, 2002.

[226] M. Naderhirn and P. Langthaler. Method and system for inspecting a surface area for material defects, June 19 2014. US Patent App. 14/113,917.

[227] E. New, D. f. E. F. Zoonotic Disease (NEZD) Division, and R. Affairs. Most common breeds of cattle in gb (nuts 1 areas), 2005.

[228] NOAA. What is geodesy? `https://oceanservice.noaa.gov/facts/geodesy.html`. [Online; accessed 2-Nov-2018].

[229] A. Noviyanto and A. M. Arymurthy. Automatic cattle identification based on muzzle photo using speed-up robust features approach. In *Proceedings of the 3rd European conference of computer science, ECCS*, volume 110, page 114, 2012.

[230] A. Noviyanto and A. M. Arymurthy. Beef cattle identification based on muzzle pattern using a matching refinement technique in the sift method. *Computers and electronics in agriculture*, 99:77–84, 2013.

[231] U. S. D. of Agriculture (USDA) Animal and P. H. I. Service. Cattle identification. `https://www.aphis.usda.gov/aphis/ourfocus/animalhealth/nvap/NVAP-Reference-Guide/Animal-Identification/Cattle-Identification`. [Online; accessed 14-November-2018].

[232] C. Olah. Understanding long-short term memory networks. `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`, 2015. [Online; accessed 12-January-2018].

[233] E. Olson. Apriltag: A robust and flexible visual fiducial system. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3400–3407. IEEE, 2011.

[234] G. Oriolo, M. Vendittelli, and G. Ulivi. On-line map building and navigation for autonomous mobile robots. In *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, volume 3, pages 2900–2906. IEEE, 1995.

[235] L. Paletta and A. Pinz. Active object recognition by view integration and reinforcement learning. *Robotics and Autonomous Systems*, 31(1-2):71–86, 2000.

[236] H. Pape. The inheritance of the piebald spotting pattern and its variation in holstein-friesian cattle and in landseer-newfoundland dogs. *Genetica*, 80(2):115–128, 1990.

[237] E. Parliament and Council. Establishing a system for the identification and registration of bovine animals and regarding the labelling of beef and beef products and repealing council regulation (ec) no 820/97. `http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32000R1760`, 1997. [Online; accessed 29-January-2016].

[238] E. Parliament and Council. Regulation (ec) no. 1760/2000 of the european parliament and of the council establishing a system for the identification and registration of bovine animals and regarding the labelling of beef and beef products and repealing council regulation (ec) no. 820/97. `https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX%3A32000R1760`, 2000. [Online; accessed 14-November-2018].

[239] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[240] M. Pastell, M. Kujala, A.-M. Aisla, M. Hautala, V. Poikalainen, J. Praks, I. Veermäe, and J. Ahokas. Detecting cow's lameness using force sensors. *Computers and Electronics in Agriculture*, 64(1):34–38, 2008.

[241] N. Pears and B. Liang. Ground plane segmentation for mobile robot visual navigation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 3, pages 1513–1518. IEEE, 2001.

[242] W. Petersen. The identification of the bovine by means of nose-prints1. *Journal of Dairy Science*, 5(3):249–258, 1922.

[243] C. Phillips and M. Rind. The effects of social dominance on the production and behavior of grazing dairy cows offered forage supplements. *Journal of Dairy Science*, 85(1):51–59, 2002.

[244] R. Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, 1999.

[245] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[246] C. Potthast and G. S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *Journal of Visual Communication and Image Representation*, 25(1):148–164, 2014.

[247] A. Poursaberi, C. Bahr, A. Pluk, A. Van Nuffel, and D. Berckmans. Real-time automatic lameness detection based on back posture extraction in dairy cattle: Shape analysis of cow with image processing techniques. *Computers and Electronics in Agriculture*, 74(1):110–119, 2010.

[248] J. Primicerio, S. F. Di Gennaro, E. Fiorillo, L. Genesio, E. Lugato, A. Matese, and F. P. Vaccari. A flexible unmanned aerial vehicle for precision agriculture. *Precision Agriculture*, 13(4):517–523, 2012.

[249] W. PS2008. Global positioning system wide area augmentation system (waas) performance standard. *Washington DC: FAA*, 2008.

[250] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[251] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

[252] P. Rajkondawar, A. Lefcourt, N. Neerchal, R. Dyer, M. Varner, B. Erez, and U. Tasch. The development of an objective lameness scoring system for dairy herds: pilot study. *Transactions of the ASAE*, 45(4):1123, 2002.

[253] P. Rajkondawar, U. Tasch, A. Lefcourt, B. Erez, R. Dyer, and M. Varner. A system for identifying lameness in dairy cattle. *Applied Engineering in Agriculture*, 18(1):87, 2002.

[254] L. Ran, Y. Zhang, Q. Zhang, and T. Yang. Convolutional neural network-based robot navigation using uncalibrated spherical images. *Sensors*, 17(6):1341, 2017.

[255] L. Rayleigh. Xii. on the resultant of a large number of vibrations of the same pitch and of arbitrary phase. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 10(60):73–78, 1880.

[256] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[257] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[258] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[259] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[260] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM, 2016.

[261] A. Robinson and F. Fallside. *The utility driven dynamic error propagation network*. University of Cambridge Department of Engineering, 1987.

[262] A. Rodríguez, J. J. Negro, M. Mulero, C. Rodríguez, J. Hernández-Pliego, and J. Bustamante. The eye in the sky: combined use of unmanned aerial systems and gps data loggers for ecological research and conservation of small birds. *PLoS One*, 7(12):e50336, 2012.

[263] R. Rojas. The backpropagation algorithm. In *Neural networks*, pages 149–182. Springer, 1996.

[264] W. Rossing. Animal identification: introduction and history. *Computers and Electronics in Agriculture*, 24(1):1–4, 1999.

[265] S. D. Roy, S. Chaudhury, and S. Banerjee. Aspect graph based modeling and recognition with an active sensor: a robust approach. 2001.

[266] S. D. Roy, S. Chaudhury, and S. Banerjee. Recognizing large 3-d objects through next view planning using an uncalibrated camera. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 276–281. IEEE, 2001.

[267] S. D. Roy, S. Chaudhury, and S. Banerjee. Active recognition through next view planning: a survey. *Pattern Recognition*, 37(3):429–446, 2004.

[268] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[269] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[270] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[271] K. Ryu. *Autonomous Robotic Strategies for Urban Search and Rescue*. PhD thesis, Virginia Polytechnic Institute and State University, 2012.

[272] P. Saeedi, P. D. Lawrence, and D. G. Lowe. Vision-based 3-d trajectory tracking for unknown environments. *IEEE transactions on robotics*, 22(1):119–136, 2006.

[273] R. C. Sandwell and T. Burghardt. Chimpanzee face detection: an automated system for images captured from natural environments. In *9th International Conference on Behaviour, Physiology and Genetics of Wildlife*. Leibnitz Institute for Zoo and Wildlife Research, 2013.

[274] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi. Divergent stereo for robot navigation: Learning from bees. In *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*, pages 434–439. IEEE, 1993.

[275] A. Savitzky and M. J. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

[276] B. Schiele and J. L. Crowley. Transinformation for active object recognition. In *Computer Vision, 1998. Sixth International Conference on*, pages 249–254. IEEE, 1998.

[277] T. C. Schroeder and G. T. Tonsor. International cattle id and traceability: Competitive implications for the us. *Food Policy*, 37(1):31–40, 2012.

[278] S. Schulter, P. Vernaza, W. Choi, and M. Chandraker. Deep network flow for multi-object tracking. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 2730–2739. IEEE, 2017.

[279] W. R. Scott, G. Roth, and J.-F. Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys (CSUR)*, 35(1):64–96, 2003.

[280] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[281] P. Sermanet, A. Frome, and E. Real. Attention for fine-grained categorization. *arXiv preprint arXiv:1412.7054*, 2014.

[282] G. Shakhnarovich, J. W. Fisher, and T. Darrell. Face recognition from long-term observations. In *European Conference on Computer Vision*, pages 851–865. Springer, 2002.

[283] C. Shanahan, B. Kernan, G. Ayalew, K. McDonnell, F. Butler, and S. Ward. A framework for beef traceability from farm to slaughter using global standards: an irish perspective. *Computers and electronics in agriculture*, 66(1):62–69, 2009.

[284] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[285] X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu, and Q. Wang. Particle swarm optimization-based algorithms for tsp and generalized tsp. *Information processing letters*, 103(5):169–176, 2007.

[286] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1297–1304. Ieee, 2011.

[287] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[288] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[289] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[290] G. Sinibaldi and L. Marino. Experimental analysis on the noise of propellers for small uav. *Applied Acoustics*, 74(1):79–88, 2013.

[291] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1442–1468, 2014.

[292] G. Smith, J. Tatum, K. Belk, J. Scanga, T. Grandin, and J. Sofos. Traceability from a us perspective. *Meat science*, 71(1):174–193, 2005.

[293] X. Song, T. Leroy, E. Vranken, W. Maertens, B. Sonck, and D. Berckmans. Automatic detection of lameness in dairy cattlevision-based trackway analysis in cow's locomotion. *Computers and electronics in agriculture*, 64(1):39–44, 2008.

[294] B. Sowell, J. Mosley, and J. Bowman. Social behavior of grazing beef cattle: Implications for management. *Journal of Animal Science*, 77(E-Suppl):1–6, 2000.

[295] D. Sprecher, D. Hostetler, and J. Kaneene. A lameness scoring system that uses posture and gait to predict dairy cattle reproductive performance. *Theriogenology*, 47(6):1179–1187, 1997.

[296] H. Steinfeld, P. Gerber, T. Wassenaar, V. Castel, M. Rosales, M. Rosales, and C. de Haan. *Livestock's long shadow: environmental issues and options*. Food & Agriculture Org., 2006.

[297] M. Stevens, I. C. Cuthill, A. M. Windsor, and H. J. Walker. Disruptive contrast in animal camouflage. *Proceedings of the Royal Society of London B: Biological Sciences*, 273(1600):2433–2438, 2006.

[298] M. Stokkeland, K. Klausen, and T. A. Johansen. Autonomous visual navigation of unmanned aerial vehicle for wind turbine inspection. In *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*, pages 998–1007. IEEE, 2015.

[299] I. Sutskever. Training recurrent neural networks. *University of Toronto, Toronto, Ont., Canada*, 2013.

[300] R. S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the seventh international conference on machine learning*, pages 216–224, 1990.

[301] L. A. Syme, G. Syme, T. Waite, and A. Pearson. Spatial distribution and social status in a small herd of dairy cows. *Animal Behaviour*, 23:609–614, 1975.

[302] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[303] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[304] M. Tadesse and T. Dessie. Milk production performance of zebu, holstein friesian and their crosses in ethiopia. *Livestock Research for Rural Development*, 15(3):1–9, 2003.

[305] L. Tai, G. Paolo, and M. Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. *arXiv preprint arXiv:1703.00420*, 2017.

[306] A. Tharwat, T. Gaber, and A. E. Hassanien. Cattle identification based on muzzle images using gabor features and svm classifier. In *International Conference on Advanced Machine Learning Technologies and Applications*, pages 236–247. Springer, 2014.

[307] A. Tharwat, T. Gaber, and A. E. Hassanien. Two biometric approaches for cattle identification based on features and classifiers fusion. *International Journal of Image Mining*, 1(4):342–365, 2015.

[308] A. Tharwat, T. Gaber, A. E. Hassanien, H. A. Hassanien, and M. F. Tolba. Cattle identification using muzzle print images based on texture features approach. In *Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014*, pages 217–227. Springer, 2014.

[309] D. M. Thome and T. M. Thome. Radio-controlled model airplanes: inexpensive tools for low-level aerial photography. *Wildlife Society Bulletin*, pages 343–346, 2000.

[310] G. Tong, X. Meng, and N. Ye. A gpu-based affine and scale invariant feature transform algorithm. *JOURNAL OF INFORMATION &COMPUTATIONAL SCIENCE*, 10(13):3991–3998, 2013.

[311] H.-K. Tsai, J.-M. Yang, Y.-F. Tsai, and C.-Y. Kao. An evolutionary algorithm for large traveling salesman problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(4):1718–1729, 2004.

[312] C. Tucker, D. Weary, and D. Fraser. Free-stall dimensions: Effects on preference and stall usage. *Journal of Dairy Science*, 87(5):1208–1216, 2004.

[313] R. Ungerfeld, C. Cajarville, M. Rosas, and J. Repetto. Time budget differences of high-and low-social rank grazing dairy cows. *New Zealand journal of agricultural research*, 57(2):122–127, 2014.

[314] J. C. van Gemert, C. R. Verschoor, P. Mettes, K. Epema, L. P. Koh, S. Wich, et al. Nature conservation drones for automatic localization and counting of animals. In *ECCV Workshops (1)*, pages 255–270, 2014.

[315] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, page 5. Phoenix, AZ, 2016.

[316] T. Van Hertem, E. Maltz, A. Antler, C. Romanini, S. Viazzi, C. Bahr, A. Schlageter-Tello, C. Lokhorst, D. Berckmans, and I. Halachmi. Lameness detection based on multivariate continuous sensing of milk yield, rumination, and neck activity. *Journal of dairy science*, 96(7):4286–4298, 2013.

[317] J. Velez, A. Sanchez, J. Sanchez, and J. Esteban. Beef identification in industrial slaughterhouses using machine vision techniques. *Spanish Journal of Agricultural Research*, 11(4):945–957, 2013.

[318] S. Viazzi, C. Bahr, A. Schlageter-Tello, T. Van Hertem, C. Romanini, A. Pluk, I. Halachmi, C. Lokhorst, and D. Berckmans. Analysis of individual classification of lameness using automatic measurement of back posture in dairy cattle. *Journal of Dairy Science*, 96(1):257–266, 2013.

[319] S. Waharte and N. Trigoni. Supporting search and rescue operations with uavs. In *Emerging Security Technologies (EST), 2010 International Conference on*, pages 142–147. IEEE, 2010.

[320] F. Wang, W. Zuo, L. Lin, D. Zhang, and L. Zhang. Joint learning of single-image and cross-image representations for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1288–1296, 2016.

[321] J. Wang and E. Olson. Apriltag 2: Efficient and robust fiducial detection. In *IROS*, pages 4193–4198, 2016.

[322] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.

[323] D. Wardrope. Problems with the use of ear tags in cattle. *Veterinary Record*, 137(26):675–675, 1995.

[324] L. Warnick, D. Janssen, C. Guard, and Y. Gröhn. The effect of lameness on milk production in dairy cows. *Journal of dairy science*, 84(9):1988–1997, 2001.

[325] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[326] R. J. Webster, C. Hassall, C. M. Herdman, J.-G. J. Godin, and T. N. Sherratt. Disruptive camouflage impairs object recognition. *Biology letters*, 9(6):20130501, 2013.

[327] P. J. Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

[328] D. Wilkes and J. K. Tsotsos. Active object recognition. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 136–141. IEEE, 1992.

[329] L. M. Wong, C. Dumont, and M. A. Abidi. Next-best-view algorithm for object reconstruction. In *Sensor Fusion and Decentralized Control in Robotic Systems*, volume 3523, pages 191–201. International Society for Optics and Photonics, 1998.

[330] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[331] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[332] C. Xu and L. Cheng. Efficient hand pose estimation from a single depth image. In *Proceedings of the IEEE international conference on computer vision*, pages 3456–3462, 2013.

[333] O. Yamaguchi, K. Fukui, and K.-i. Maeda. Face recognition using temporal image sequence. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 318–323. IEEE, 1998.

[334] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song. Recent advances and trends in visual tracking: A review. *Neurocomputing*, 74(18):3823–3831, 2011.

[335] Y. Yang, J. Yang, J. Yan, S. Liao, D. Yi, and S. Z. Li. Salient color names for person re-identification. In *European conference on computer vision*, pages 536–551. Springer, 2014.

[336] K. Yao, T. Cohn, K. Vylomova, K. Duh, and C. Dyer. Depth-gated lstm. *arXiv preprint arXiv:1508.03790*, 2015.

[337] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 731–738. IEEE, 2011.

[338] B. Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[339] G. Yu and J.-M. Morel. A fully affine invariant image comparison method. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1597–1600. IEEE, 2009.

[340] G. Yu and J.-M. Morel. Asift: An algorithm for fully affine invariant comparison. *Image Processing On Line*, 1:11–38, 2011.

[341] H. Yu, S. Lin, J. Wang, K. Fu, and W. Yang. An Intelligent Unmanned Aircraft System for Wilderness Search and Rescue.

[342] L. Zaorálek, M. Prilepok, and V. Snášel. Cattle identification using muzzle images. In *Proceedings of the Second International Afro-European Conference for Industrial Advancement AECIA 2015*, pages 105–115. Springer, 2016.

[343] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[344] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard. Deep reinforcement learning with successor features for navigation across similar environments. *arXiv preprint arXiv:1612.05533*, 2016.

[345] J. Zhang, L. Tai, J. Boedecker, W. Burgard, and M. Liu. Neural slam. *arXiv preprint arXiv:1706.09520*, 2017.

[346] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *European conference on computer vision*, pages 864–877. Springer, 2012.

[347] L. Zhang, T. Xiang, and S. Gong. Learning a discriminative null space for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1239–1248, 2016.

[348] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[349] S. Zhang, D. Wei, M. Q. Huynh, J. X. Quek, X. Ma, and L. Xie. Model predictive control based dynamic geofence system for unmanned aerial vehicles. In *AIAA Information Systems-AIAA Infotech@ Aerospace*, page 0675. 2017.

[350] X. Zhang, H. Luo, X. Fan, W. Xiang, Y. Sun, Q. Xiao, W. Jiang, C. Zhang, and J. Sun. Alignedreid: Surpassing human-level performance in person re-identification. *arXiv preprint arXiv:1711.08184*, 2017.

[351] X. Zhang, H. Xiong, W. Zhou, W. Lin, and Q. Tian. Picking deep filter responses for fine-grained image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1134–1142, 2016.

[352] Y. Zhang, B. Li, H. Lu, A. Irie, and X. Ruan. Sample-specific svm learning for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1278–1287, 2016.

[353] K. Zhao and R. Jurdak. Understanding the spatiotemporal pattern of grazing cattle movement. *Scientific reports*, 6:31967, 2016.

[354] R. Zhao, W. Ouyang, and X. Wang. Learning mid-level filters for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 144–151, 2014.

[355] L. Zheng, Z. Bie, Y. Sun, J. Wang, C. Su, S. Wang, and Q. Tian. Mars: A video benchmark for large-scale person re-identification. In *European Conference on Computer Vision*, pages 868–884. Springer, 2016.

[356] L. Zheng, Y. Yang, and A. G. Hauptmann. Person re-identification: Past, present and future. *arXiv preprint arXiv:1610.02984*, 2016.

[357] Z. Zheng, L. Zheng, and Y. Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3754–3762, 2017.

[358] Z. Zheng, L. Zheng, and Y. Yang. A discriminatively learned cnn embedding for person reidentification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(1):13, 2018.

[359] S. K. Zhou and R. Chellappa. Beyond one still image: Face recognition from multiple still images or a video sequence. *Face processing: advanced modeling and methods*, pages 547–567, 2005.

[360] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE, 2017.

Appendix A

Simulation Environment

To operate outside of the constraints imposed by experimentation on real data and assist with the synthesis of appropriate realistic training, testing and validation data, a comprehensive simulation environment was created. The environment operates within Gazebo[1], an open-source multi-robot simulation framework [164]. Crucially, Gazebo integrates naturally with ROS [251] for asynchronous robot and simulator control, for which, the implementation of this thesis – operating on-board embedded systems situated on the UAV – is founded within, both in simulation and in reality. Also a crucial component of the implemented simulation environment is ROS package Hector Quadrotor[2] [212] for realistically simulating the physical properties of a UAV; it is fullly discussed later in Section A.1.

The simulation environment itself (as illustrated in Figure A.1) consists of a ground plane with a realistic, infinitely-repeating grass texture, moving clouds within a sky box and a simulated sun light source that casts physically correct shadows on models placed within the environment. Individual cattle models are also placed within the environment and are discussed further in Section A.2.

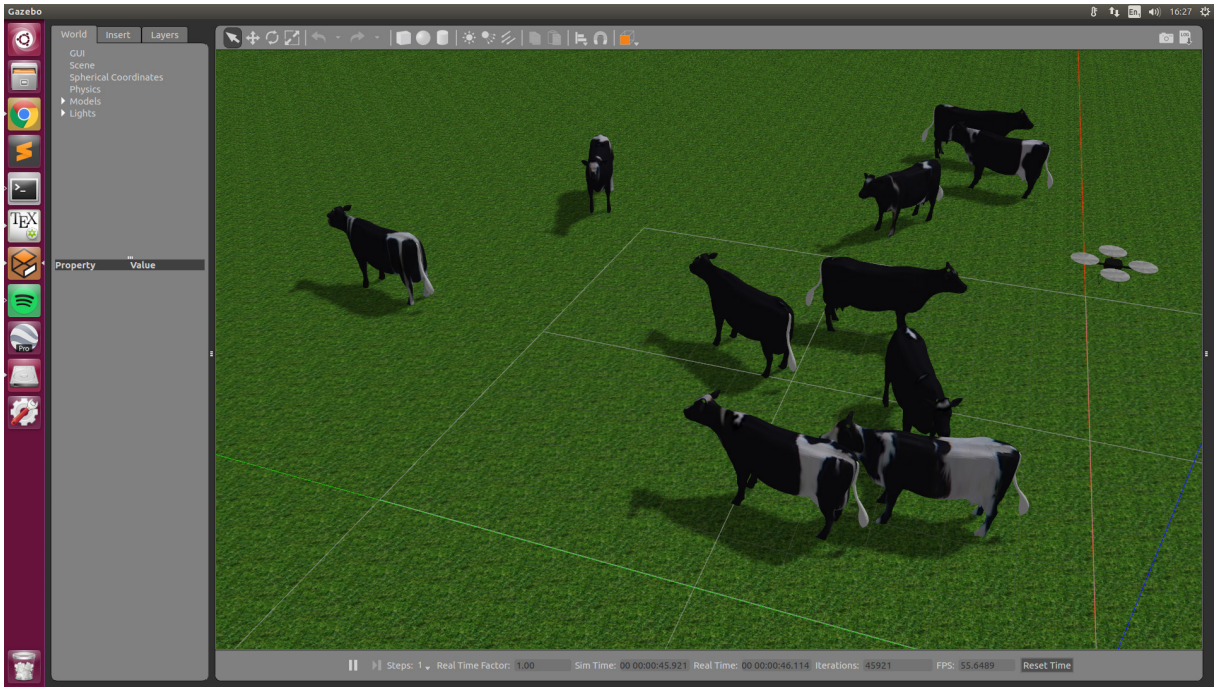
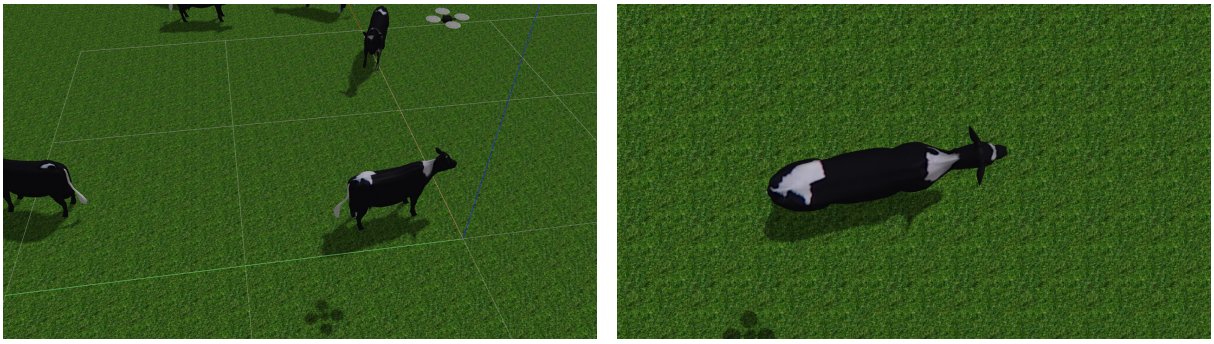


Figure A.1: ***Gazebo-Based Simulation Environment.*** *Screenshot of the Gazebo simulation environment utilised for model verification and the synthesis of training data, etc. Featured are 10 randomly placed and oriented cow models with individually-unique textures alongside the simulated UAV agent. Also visible are realistic shadows created by the sun light source.*

[1] Gazebo simulation framework: http://gazebosim.org/
[2] Hector quadrotor ROS package: http://wiki.ros.org/hector_quadrotor

A.1 UAV Model

The complete simulation environment includes the utilisation of a fully simulated UAV. The utilised ROS package, named Hector Quadrotor [212], consists of sub-packages related to the modelling, control and simulation of quadrotor UAV systems. Not only can realistic UAV dynamics be modelled, the package also includes components for simulating fully customisable UAV-mounted sensory instruments (e.g. camera, LiDAR, depth sensor) via custom XML definitions[3]. In the case here, a two-axis (pitch, yaw) gimbal camera mounted on the underside of the aircraft is simulated. An example of this simulated view is given in Figure A.2 at a resolution of 640×480 pixels with horizontal FoV 60°. Within simulated experiments conducted in this thesis, the control elements imposed by the physics of the simulated UAV via the Hector Quadrotor package are abstracted from to retain simplicity and provide a fundamental proof-of-concept. This is since, in reality the API implemented on-board the utilised UAV flight platform (DJI Matrice 100, see Section 7.2) for real outdoor flight experiments also abstracts specific control commands – as may be controlled by a flight position PID controller – away from the user. They are instead issued in the form of direct position commands relative to a user set reference frame aligned with the earth's ENU axis. Consequently, within the simulation environment, the simulated UAV is regarded as a completely static object in 3D space (with no corresponding physics simulation) and is teleported from position to position per iteration or timestep, given some action to fulfil.



(a) Simulated UAV *(b) Camera feed*

Figure A.2: ***Simulated UAV and Camera Feed.*** *Rendered images of (a): the Gazebo-based simulated UAV utilising the Hector Quadrotor ROS package [212] and (b): the corresponding simulated* 640×480 *camera image feed from the simulated flight platform with the camera pointed directly downwards.*

A.2 Cow Model

To obtain a good graphical approximation of a Holstein Friesian cow, a suitably detailed 3D cow model was selected and purchased from TurboSquid; a popular 3D model marketplace website[4]. Free and open-source 3D graphics manipulation software Blender[5] [33] was then utilised to alter the model as required. In particular, an orthogonal projection was employed to generate the *uv*-map (illustrated in Figure A.3) used later for texture generation and projection. Additionally, the model was aligned with the Gazebo reference frame and the origin placed at the centre of the object's bottom-most xy plane. Finally, Blender was used to proportionally scale the model to the correct unit length (in metres). The height value of 1.27m was selected based on the typical height of a middle-aged female Holstein Friesian [138]. All other axes (x, y) were scaled linearly to retain model proportionality. Individual textures for this three dimensional cow model are manually generated in appropriate photo editing software. This

[3] Xacro (XML Macros): `http://wiki.ros.org/xacro`
[4] Turbosquid: 3D model marketplace: `https://www.turbosquid.com/`
[5] Blender: open-source 3D modelling software: `https://www.blender.org/`

process consists of drawing desired visual features over the *uv*-map template that can be seen in Figure A.3. One distinct disadvantage of *uv*-mapping in this simple form of top-down, orthogonal projection is the fact that one cannot describe differing visual details for overlapping model polygons/areas (e.g. the sides and underside of the cow).

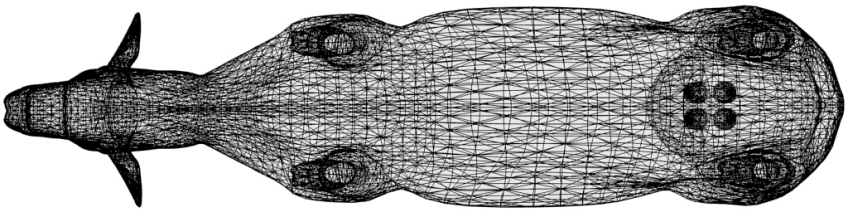


Figure A.3: ***Cattle UV Map.*** *Planar, top-down uv-map projection for the utilised cow 3D model.*

Appendix B

GPS Coordinate Fulfilment

This appendix describes the necessary steps in fulfilling a requested GPS coordinate using the DJI M100 UAV flight platform, as is frequently required throughout the conduction of experiments in Chapter 7. This is a consequence of the DJI API not supporting the ability to directly command GPS positions at the time of writing. Instead, position commands are issued via local position offsets in metres with respect to a programatically-set local ENU reference frame. Consequently, in order to fulfil a target GPS coordinate, it must be converted into that same frame. This is achieved by converting the target GPS coordinate into the static ECEF reference frame, then converting that coordinate into the local ENU frame. Equally, the same process is performed on the agent's current GPS position and the resulting local positions are compared to find an offset. The implementation of this process – following common standards established in literature [25, 137] – is fully described as follows. Note also that all GPS coordinates refer to the WGS-84 GCS standard [63].

To remind the reader, ECEF defines the earth's static Cartesian reference frame independent of the earth's rotation with the point $(0,0,0)$ defining the earth's centre of mass [56]. The x-axis of ECEF is aligned with latitudinal and longitudinal coordinates 0° and 0°, respectively, and z is aligned with true north. To complete the definition, the y-axis also has latitudinal value 0° but is orthogonal to both x, y in a right-handed coordinate system. Converting a WGS-84 GPS coordinate into ECEF begins with converting the coordinate tuple $A = (latitude, longitude, altitude)$ into appropriate radian counterparts:

$$\begin{aligned} \phi &= A_{latitude} \cdot \frac{\pi}{180}, \\ \lambda &= A_{longitude} \cdot \frac{\pi}{180}, \\ h &= A_{altitude}. \end{aligned} \tag{B.1}$$

Next, this (ϕ, λ, h) tuple can be converted into ECEF coordinates via:

$$\begin{aligned} A_x &= (N(\phi) + h) cos\phi cos\lambda, \\ A_y &= (N(\phi) + h) cos\phi sin\lambda, \\ A_z &= \left(\frac{a^2}{b^2} N(\phi) + h \right) sin\phi, \end{aligned} \tag{B.2}$$

where

$$N(\phi) = \frac{a^2}{\sqrt{a^2 cos^2\phi + b^2 sin^2\phi}} = \frac{a}{\sqrt{1 - e^2 sin^2\phi}} \tag{B.3}$$

with

$$e^2 = 1 - \frac{a^2}{b^2} \tag{B.4}$$

where $a = 6378137.0\ m$, $b = 6356752.314245\ m$ and e^2 denote the earth's equatorial (semi-major axis) and polar radii (semi-minor axis) and the square of eccentricity of the earth ellipsoidal model, respectively.

Subsequently, this ECEF coordinate (A_x, A_y, A_z) is converted into a ENU-based reference frame defined by another WGS-84 GPS coordinate denoting the origin. In practise, this is chosen to be the GPS coordinate recorded prior to aircraft takeoff defined geodetically by $O = (\phi, \lambda, h)$ (note that the tuple is assumed to have already been converted into radians for the latitudinal and longitudinal components). This origin O for the ENU frame is then expressed in the ECEF reference frame via equation B.2 resulting in (O_x, O_y, O_z). The positional difference is then found via:

$$\begin{aligned} \Delta x &= A_x - O_x, \\ \Delta y &= A_y - O_y, \\ \Delta z &= A_z - O_z. \end{aligned} \tag{B.5}$$

The original GPS coordinate $A_{ENU} = (x_E, y_N, z_U)$ can then be resolved in the defined ENU reference frame with respect to the origin O via:

$$\begin{bmatrix} x_E \\ y_N \\ z_U \end{bmatrix} = \begin{bmatrix} -sin\phi & cos\phi & 0 \\ -cos\phi sin\lambda & -sin\lambda sin\phi & cos\lambda \\ cos\lambda cos\phi & cos\lambda sin\phi & sin\lambda \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \tag{B.6}$$

Finally, to determine the local position offset of two GPS coordinates B, C as required by the M100 flight platform's API, one would convert both geodetic coordinates to the local ENU frame via equations B.2 and B.6 and then determine per-dimension error (similarly to equation B.5).